BAYESIAN ADAPTATION AND COMBINATION OF DEEP MODELS FOR AUTOMATIC SPEECH RECOGNITION

A Dissertation
Presented to
The Academic Faculty

By

Zhen Huang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
August 2017

BAYESIAN ADAPTATION AND COMBINATION OF DEEP MODELS FOR AUTOMATIC SPEECH RECOGNITION

Approved by:

Dr. Chin-Hui Lee, Advisor
Committee Chair
Professor, School of ECE
Georgia Institute of Technology

Dr. Biing-Hwang (Fred) Juang
Professor, School of ECE
Georgia Institute of Technology

Dr. Mark A. Clements
Professor, School of ECE
Georgia Institute of Technology

Dr. Geoffrey Ye Li
Professor, School of ECE
Georgia Institute of Technology

Dr. Sabato Marco Siniscalchi
Associate Professor, Department of Computer and Telecommunications Engineering
University of Enna Kore

Date Approved: May 5th, 2017

Dedicated to my parents, my wife and my child

for their boundless

love and support

ACKNOWLEDGMENTS

Like a miracle, 5 years ago when I just got to the US, I was so nervous about the survival as a Ph.D. student; now I am doing exciting cutting-edge research in a renowned company. It is my advisor, Prof. Chin-Hui Lee, who makes that happen. I owe my most sincere gratitude to him. Along the journey of my research pursuit, it is his visionary thought and insightful guidance that help me walk through hard times. It is my best fortune to have been working with him and he has been and will always be a role model through my life.

Thanks to the professors who spend their precious time to serve on my dissertation committee: Prof. Biing-Hwang (Fred) Juang, Prof. Mark A. Clements, Prof. Geoffrey Ye Li and Prof. Sabato Marco Siniscalchi. I have to specially thank Prof. Biing-Hwang (Fred) Juang for showing me the fantastic world of speech processing where I find my passion, Prof. Mark A. Clements for his valuable suggestions and comments during the preparation of my dissertation and Prof Sabato Marco Siniscalchi for being my most important collaborator and best friend. I also want to express my gratitude to Prof. Guotong Zhou. The GT-SJTU dual-master program she initiated and operated changed my life.

During the pursuit of my Ph.D. at Georgia Tech these years, I owe my thanks to many people. Thanks to You-Chi Cheng for his selfless help especially during my early years into the Ph.D. program. Thanks to Chao Weng for his collaboration in research and the tremendous help in my professional career. Thanks to Kehuang Li for being my most trustable colleague and friend. I would also extend my thanks to other friends in Georgia Tech: I-Fan Chen, Wei Li, Sicheng Wang, Zhong Meng, Sam Li, Lijun Zhu, Zhen Wang, Yuting Hu, Taizhi Liu, Yan Li, Xinjun Dong, Yu Liu, Rui Chen, Zhibo Yuan, Prof. Ji Wu, Jiadong Wu, Jun Zou, Qingsong Wen, Zhaoyu Wang, Zhenhua Yu, Yihai Fang, Huayu Li and Chong Han. Thanks to Pat Dixon, Raquel Plaskett, Jennifer Lunsford, Daniela Staiculescu, and Tasha Torrence for their great administrative support.

I am very fortunate to work as an intern in Microsoft and meet excellent researchers there. Thanks to Chaojun Liu for his great mentorship; thanks to Dong Yu for his valuable advice and collaboration; thanks to Yong Zhao for hosting me during that beautiful summer; thanks to Yifan Gong for his kind tolerance when I made mistakes; thanks to Qi Miao, Jian Xue, Kshitiz Kumar, Ye Tian, Yan Huang, Kaustubh Kalgaonkar, Yongqiang Wang for the effective collaboration. Special thanks go to Jinyu Li for leading me into the field of speech recognition and the guidance all the way along my research path.

I also owe great acknowledge to my colleagues in Apple Siri. Thanks to Xiaodan Zhuang and Daben Liu for mentoring me on such an interesting research topic during my internship and their patient guidance in my current job. Thanks to Bing Zhang for his selfless help as a friend. Thanks to Hans Geuns-Meyer, Sue Tranter, Chris Smolinski, Liz Brooks, Arnab Ghoshal and Bing Zhang for proofreading my dissertation. Thanks to all my colleagues in Apple Siri making Siri family to me.

Finally, I am greatly indebted to my parents for their boundless love and support and I want to express my deepest gratitude to my wife, Hang, who has always been and will always be with me in my journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The goal of this dissertation is to explore issues related to Bayesian adaptation and combination of deep models in the field of automatic speech recognition (ASR). At the time of writing this dissertation, research on ASR has been ongoing for almost 70 years. Today's ASR systems have come a long way since the first single-speaker, isolated digit recognizer made at Bell Laboratories [1] in the 1950s, leveraging many advanced machine learning technologies. In particular, recent advances in deep neural networks (DNNs) [2] have spurred new research efforts in DNN based acoustic and language modeling, which are the two key components of modern ASR systems.

However, the common assumption that training and testing data belong to the same feature space and have the same distribution overlooks the underlying discrepancies between ideal and realistic conditions. Therefore, performance of a modern ASR system in the field may not reflect the performance measured during system design. For example, we have learned from the outcome of the 2014 REVERB CHALLENGE [3] that a traditional Gaussian Mixture Model (GMM) / Hidden Markov Model (HMM) ASR system trained and tested under clean condition attains a Word Error Rate (WER) of 3.50%. However, a severe drop in performance is observed by testing the same ASR system on reverberant speech. With reverberant speech an average WER of 42.80% was measured. Replacing the GMM with a DNN did not produce any significant improvement, but rather, a further small drop in the ASR performance was observed. This same phenomenon is observable when recognizing speech uttered by a non-native speaker using an ASR system trained only on native speakers' data. In the Resource Management task [4], WER drops to 34.90% on non-native speech from the initial 3.60% for native speakers which matched the training conditions.

The fact that ASR systems exhibit suboptimal performance when there is a mismatch

between training and test conditions indicates a lack of robustness. The mismatch can be a combination of several different conditions, including: (i) ambient noise, such as background speech, and background noise; (ii) channel variations due to different microphones and room reverberation; (iii) dialects and accents. Many approaches have been proposed over the years to solve these issues. Model adaptation techniques are among the most important ones to tackle the mismatch problem. In the model adaptation, the original set of acoustic models is mapped to a transformed set of models that better match the observed utterances using a small but representative amount of data (a.k.a., adaptation data) [5, 6]. Speaker adaptation, a particular variant of model adaptation has become a popular research topic since the 90s. Speaker adaptation seeks to adapt a general ASR system to a specific speaker with a limited amount of data from that speaker. With the prevalence of ASR facilitated personal assistants such as *Siri* and *Google Now*, the function of speaker adaptation to provide personalized user experience attracts more and more attention. Another important method for enhancing the performance of ASR systems is system combination, where the goal is to improve accuracy by leveraging complementary information from different systems to produce a "super-system" which is better than each contributing individual system. [7, 8]. There are a lot of approaches to building ASR systems, and those approaches can differ in the feature extraction method, classification approach, and training algorithm. They often use complementary information about speech, but the attempt to preserve one part of the information often leads to the loss of another. As a consequence, each individual ASR approach could only attain a different degree of success. System combination has thus been used to improve the ASR performance. More discussion about model adaptation and system combination can be found in Chapter 2.

This dissertation leverages Bayesian learning to address the model adaptation and system combination problem when only a small amount of target speaker data is available. The focus will be on the issues arising when model adaptation and system combination

are applied to deep learning based, state-of-the-art ASR systems. Detailed discussions about deep learning, model adaptation, and system combination can be found in Chapter 2. In this chapter, the key technology components that make an ASR system work will be briefly described.

1.1 Automatic Speech Recognition System

The task of speech recognition is to recognize the word sequence $\mathbf{W}$ from the signal $\mathbf{X}$. This is actually a decision problem: Based on the information in $\mathbf{X}$ we attempt to make the best inference about $\mathbf{W}$ that is embedded in $\mathbf{X}$. A speech signal is usually featured by uncertainty, variability, lack of determinism, and stochasticity which makes the statistical pattern-matching paradigm a natural choice for formulating and solving the ASR problem. If the joint distribution $p(\mathbf{W}, \mathbf{X})$ is specified, the Bayes' decision rule [9] can be implemented as:

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} p(\mathbf{W}, \mathbf{X}), \tag{1.1.1}$$

where $\hat{\mathbf{W}}$ is the recognized words. However, it is unlikely that we have complete knowledge to specify the joint distribution $p(\mathbf{W}, \mathbf{X})$. For real-world speech recognition, $p(\mathbf{W}, \mathbf{X})$ is decomposed into two components:

$$p(\mathbf{W}, \mathbf{X}) = p(\mathbf{X}|\mathbf{W})p(\mathbf{W}) \tag{1.1.2}$$

where $p(\mathbf{X}|\mathbf{W})$ is known as the acoustic model (AM) and $p(\mathbf{W})$ the language model (LM). The AM $p(\mathbf{X}|\mathbf{W})$ is used for evaluating the likelihood of the observation $\mathbf{X}$ given a word sequence, and the LM $p(\mathbf{W})$ computes the probability of the word sequence $\mathbf{W}$ [10]. Given a training set, we can estimate parameters of both the AM and the LM, $\hat{\Lambda}$ and $\hat{\Gamma}$, respectively. Then the estimated parameters are plugged into the maximum a posteriori decision rule (Eq. 1.1.1) as:

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} p_{\hat{\Lambda}}(\mathbf{X}|\mathbf{W})p_{\hat{\Gamma}}(\mathbf{W}), \tag{1.1.3}$$

Figure 1.1.1. A typical block diagram of a ASR system

The block diagram of ASR system is shown in Figure 1.1.1. The three main components are the feature extractor, the acoustic model and the language model. Next we will briefly discuss feature extraction as well as acoustic and language modeling for the ASR system.

1.1.1 Feature extraction

A speech signal can be considered as the output of a slowly time-varying linear system driven by an excitation signal with the practical assumption of short-term stationarity. There are several ways of representing the time-varying speech signals, for example, via the state of the speech production source – the vocal cords, via a spectral representation and via a parametrization of the spectral activity based on the model of speech production. [11]. Among them, the spectral shape is deemed as the most important feature representation for ASR. The cepstrum is a common choice of spectral shape representation, which is the inverse Fourier transform (IFT) of the log-power-spectrum. Mel-Frequency Cepstral Coefficients (MFCCs) [12] is the most widely used cepstral feature. Triangular filters evenly distributed on the Mel-frequency scale reflect the intent to mimic the human auditory system in response to acoustic/auditory stimulation. Log-energy is computed at the output of each filter and the final MFCCs are the discrete cosine transform (DCT) of these filter outputs. Besides spectral shape based features, researchers are also exploring the possibility of directly using raw time-domain signals [13] as input features for DNN.

1.1.2 Acoustic Models

The central component of the acoustic model in a modern ASR system is the Hidden Markov Model (HMM) [11, 14]. The adoption of HMM [14, 15] is one of the greatest breakthroughs in the field of ASR. An HMM is a statistical model for generating

sequences of symbols (in speech recognition, speech frames). A context-dependent phoneme, e.g a triphone or a quinphone, is represented by an HMM. States of HMMs are aligned with speech frames using the forward-backward or the Viterbi algorithm [14].

In a traditional GMM based HMM system, the state emission probability is modeled by a mixture of Gaussians,

$$f(\mathbf{o}_t|s_j) = \log \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{o}_t|s_j), \tag{1.1.4}$$

where $\mathbf{o}_t$ is the feature frame of speech signal at time t; s_j denotes the jth HMM state. A decision tree is then used to tie together model parameters of those triphone/quinphone HMM states (senones) in order to control the model complexity.

The state-of-the-art ASR systems replace the GMM part of a GMM-HMM system with a deep neural network to leverage the strong modeling power of deep models. The replacement is straight-forward, but it comes with a huge change in the model assumption that brings both pros and cons. We will elaborate on this in Chapter 2.

1.1.3 Language Model

The language model (LM) provides a way to estimate the probability of a possible word sequence. The n-gram LM is still the most commonly adopted LM. The probability of the next word given its history in an utterance is estimated by the maximum likelihood principle in the n-gram LM. However, one critical issue for the n-gram LM is how to deal with data sparseness which causes unseen words to be assigned zero probabilities. The solution is to use smoothing techniques including Jelinek-Mercer interpolation [16], Katz backoff [17], Witten-Bell smoothing [18], absolute discounting and Kneser-Ney smoothing [19]. The state-of-the-art language modeling performance [20] is achieved by using the recurrent neural network (RNN) based LM introduced in [21] where the probability of the next word is provided by the output of a RNN.

1.2 Scientific Goals

As mentioned in the beginning of this chapter, even with the powerful DNN model, the context dependent deep neural network hidden Markov models (CD-DNN-HMMs) system still suffers losses due to the mismatch between training and testing conditions. In this dissertation, we try to tackle the mismatch problem for deep model based ASR systems by applying Bayesian adaptation and combination.

Bayesian approaches obtained great success in the adaptation of the traditional context dependent Gaussian mixture model hidden Markov models (CD-GMM-HMMs). Bayesian point estimation techniques such as maximum a posteriori (MAP) [5], structured MAP [22] and variational Bayesian methods [23] are all successful examples of Bayesian being applied to CD-GMM-HMM acoustic model adaptation. Furthermore, MAP and SMAP have been shown to be promising to address data scarcity issues while providing a highly desirable asymptotic performance as the number of adaptation sentences increases toward infinity. However, CD-DNN-HMM's discriminative nature makes it difficult to apply Bayesian based adaptation techniques on it. What's more, the huge number of CD-DNN-HMM parameters makes adaptation of deep models a challenging task, particularly when the adaptation data is limited.

The objective of this dissertation is to deploy a Bayesian adaptation/combination framework for deep model based ASR systems to combat degradation of recognition accuracy, which is typically observed under potentially mismatched conditions between training and testing. This dissertation addresses the problem in three directions.

The first direction is to perform Bayesian adaptation directly on the discriminative DNN models. To directly adapt the DNN models, maximum a posteriori estimation is employed in the manner of regularization in the DNN updating formula. For supervised batch adaptation, we built the adaptation technology based on MAP estimation of the parameters of an augmented linear hidden network (LHN) at the adaptation time. The proposed MAP adaptation scheme can provide a substantial relative WER reduction from

an already-strong speaker independent CD-DNN-HMM baseline and can consistently outperform conventional transformation based adaptation schemes. We then apply the proposed MAP adaptation scheme on the unsupervised online adaptation to make the technique applicable to practical situations. To deal with the data scarcity problem which becomes more severe in online adaptation, we reduce the parameter number by applying the MAP adaptation framework on the activation function parameters only, instead of adapting the augmented LHN parameters. The proposed Bayesian framework is adaptive in nature and suitable for performing iterative learning. Promising online adaptation results are obtained. Following the thinking of adding regularization in adaptation, a hierarchical adaptation technique was proposed through multi-task learning (MTL) [24] devising regularization provided by the auxiliary tasks. By adding the auxiliary architecture to the original DNN and performing MTL with the secondary tasks, we improve the learning ability of the original DNN structure, thus enhancing the discrimination power of the DNN models with limited adaptation data.

In the second direction, we try to cast the DNN into a generative framework to better leverage Bayesian techniques. Although we were able to define and estimate prior distribution to perform MAP adaptation for the CD-DNN-HMMs with the above-mentioned Bayesian adaptation methods, the discriminative nature of DNN models hampered our goal to replicate the nice features of the MAP/SMAP solutions developed for the generative CD-GMM-HMMs. We then cast the DNN into a generative framework to facilitate the employment of classical ML and MAP techniques. We focused on the adaptation of weights and biases in the last DNN affine transformation layer and devise a maximum likelihood (ML) estimator to perform model adaptation. Next, a MAP adaptation scheme was formulated by incorporating prior information obtained by applying the proposed ML approach to the training data. Experimental results show that the proposed approach gives better results than conventional transformation based approaches applied to the same parameters. Unfortunately, MAP is not always able to

further improve the adaptation results. To regain the theoretical advantages of the MAP/SMAP solutions, we go an indirect way by converting the discriminative CD-DNN-HMMs to the generative CD-GMM-HMMs. Using the bottleneck (BN) features [25–27] (BN concept was first introduced by [25]. [26] and [27] are successful applications to ASR) derived from the CD-DNN-HMMs, we were able to build deep BN-CD-GMM-HMMs (BN feature based CD-GMM-HMMs) with performance comparable to CD-DNN-HMMs. Then, the Bayesian adaptation was done by applying the original MAP/SMAP techniques to the BN-CD-GMM-HMMs. Experimental results show the effectiveness of the proposed indirect Bayesian method and also demonstrate the ability of DNN to serve as a bridge function between data distribution and the GMM based decision model [28].

In the third direction, we employed a hierarchical Bayesian system combination technique to further enhance the adaptation performance by leveraging the complementarity of the discriminative and generative adaptive models. The CD-DNN-HMMs adapted with MAP-LHN in the direct manner and the BN-CD-GMM-HMMs adapted with SMAP in the indirect manner utilize different model assumptions and are adapted with different techniques. As a result, their potential to be complementary is promising. A novel system combination scheme is proposed to leverage this complementarity to improve the adaptation performance. In the proposed system combination scheme, per-speaker and per-senone (tied triphone HMM state) [29] combination weights are learned from the adaptation data through the hierarchical Bayesian approach. The system combination is performed in a dynamic way, exploiting the temporal nature of HMMs. Thus we are able to support real-time multi-stream speech recognition instead of combining systems using traditional methods in a static manner. Experimental results show that the proposed method provides significantly improved WER reductions on the already-adapted baseline deep models.

1.3 Contributions

- A Bayesian adaptation framework is proposed for direct adaptation of DNN based acoustic model using MAP estimation on the neural network's transformation parameters. The proposed MAP adaptation scheme provides a substantial relative word error rate reduction from a strong speaker independent CD-DNN-HMM baseline on the WSJ task.

- A Hierarchical adaptation framework is proposed for a direct adaptation of transformation parameters of deep neural networks which utilizes the multitask learning technique. Leveraging auxiliary information provided by the secondary tasks, the proposed multitask adaptation scheme covers a larger acoustic space than what can be covered using only the primary task, and thus providing consistent improvements over the conventional adaptation scheme that does not use multitask learning on the WSJ task.

- The direct MAP adaptation framework is extended from the adaptation of the neural network's transformation parameters to the adaptation of its activation function parameters in order to further reduce the number of parameters. The framework is applied to more practical and popular tasks of unsupervised online adaptation. Experimental results on the large-scale spontaneous conversational speech task Switchboard show the feasibility and effectiveness of the proposed approach.

- Indirect adaptation methods of deep neural network based acoustic models are proposed. By converting the discriminative CD-DNN-HMMs to the generative CD-GMM-HMMs through bottleneck (BN) features, we are able to leverage the nice features of the classical MAP/SMAP adaptation techniques originally designed for the generative CD-GMM-HMMs model in the adaptation of deep models. More than 10% WER reduction is observed on the WSJ task by using the proposed indirect Bayesian adaptation technique.

- By leveraging the complementarity of the discriminative and generative adaptive models, a hierarchical Bayesian system combination technique is employed to further enhance the adaptation performance. In the proposed system combination scheme, per-speaker and per-senone combination weights are learned from the adaptation data through the hierarchical Bayesian approach. We perform system combination in a dynamic way, exploiting the temporal nature of HMMs, thus we are able to support real-time multi-stream speech recognition instead of combining systems using traditional methods in a static manner. Experimental results on the Switchboard task show the effectiveness of the proposed method with significantly improved WER reductions on the already adapted baseline deep models.

This dissertation is organized as follows: In Chapter 2 we talk about the backgrounds and challenges of deep learning, model adaptation, and system combination in the field of ASR. In Chapter 3 and 4 we present our efforts on direct Bayesian adaptation on discriminative DNN models by performing adaptation on transformation parameters and activation function parameters, respectively. In Chapter 5, we perform Bayesian adaptation by casting the DNN to a generative model. Chapter 6 presents a Bayesian system combination approach to leverage the complementary nature of the discriminative and generative adaptive models introduced in Chapter 4 and 5, respectively. Chapter 7 concludes the study of this dissertation.

CHAPTER 2

BACKGROUNDS

In this chapter, Section 2.1 first gives an overview of the developments and problems introduced by deep learning in the field of ASR. Model adaptation techniques for both GMM and DNN base ASR systems are introduced in Section 2.2. Finally, we will review system combination in the last section.

2.1 Deep Learning in Automatic Speech Recognition

Recent success in adopting the CD-DNN-HMMs [30] has demonstrated significant performance improvements in the field of automatic speech recognition (ASR). Compared with the conventional context dependent Gaussian mixture model hidden Markov models (CD-GMM-HMMs) [14], the adoption of deep neural networks has shown huge gains in recognition accuracy in various tasks and datasets [2].

Deep structured learning, more commonly called deep learning has recently emerged as a new area of machine learning research [31, 32]. Historically, the concept of deep learning originates from artificial neural networks (ANNs) [33]. The earliest and most famous deep neural network (DNN) is a traditional feed-forward artificial neural network, a.k.a. multilayer perceptron (MLP), with many hidden layers. By 1965, Ivakhnenko et al. [34] had built an 8-layer neural network and proposed an algorithm for deep models' optimization [35]. The term deep learning was introduced in 1986 by Rina Decther [36] for the machine learning community. In the artificial neural network community, Aizenberg introduced it in 1999 [37]. However, back-propagation (BP) [38–41], the well-known algorithm for learning the parameters of neural networks, did not work well for learning deep networks with a large number of hidden layers due to its susceptibility of getting trapped in poor local optima or saddle points when started from random initialization. There was also a lack of computing power needed to deal with the huge

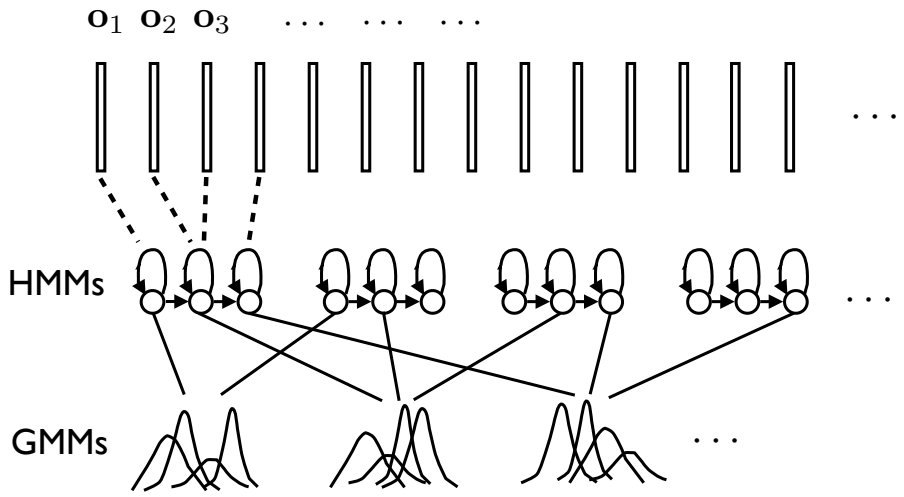

Figure 2.1.1. Context dependent Gaussian mixture model hidden Markov models (CD-GMM-HMMs)

number of parameters introduced by the deep architecture. In 2006, a pre-training method was proposed for a better initialization of parameters by growing the neural network layer by layer without using the label information. Treating each pair of layers in the network as a restricted Boltzmann machine (RBM), each layer of the neural network can then be trained using an objective criterion called contrastive divergence [42, 43]. With the pre-training method and the progress of computing techniques, especially the use of general purpose graphic processing units (GPGPU) to accelerate the BP algorithm, deep learning began to emerge as a powerful learning paradigm in various fields such as computer vision [44, 45], automatic speech recognition (ASR) [2, 30], and natural language processing [46].

In the field of ASR, the deep architecture of CD-DNN-HMMs [30] is built by replacing the shallow ANN in the ANN-HMM hybrid models (known as connectionist systems) [47] with a DNN that is much deeper. It can also be seen as replacing the GMM part from the traditional CD-GMM-HMMs with a DNN. CD-DNN-HMMs have demonstrated significant performance improvements over the conventional CD-GMM-HMMs in various tasks and datasets [2]. The reason why DNNs surpass GMMs in acoustic modeling is still an open question. In [48] researchers find that the gain of DNN almost entirely comes from DNN's feature vectors that are concatenated from

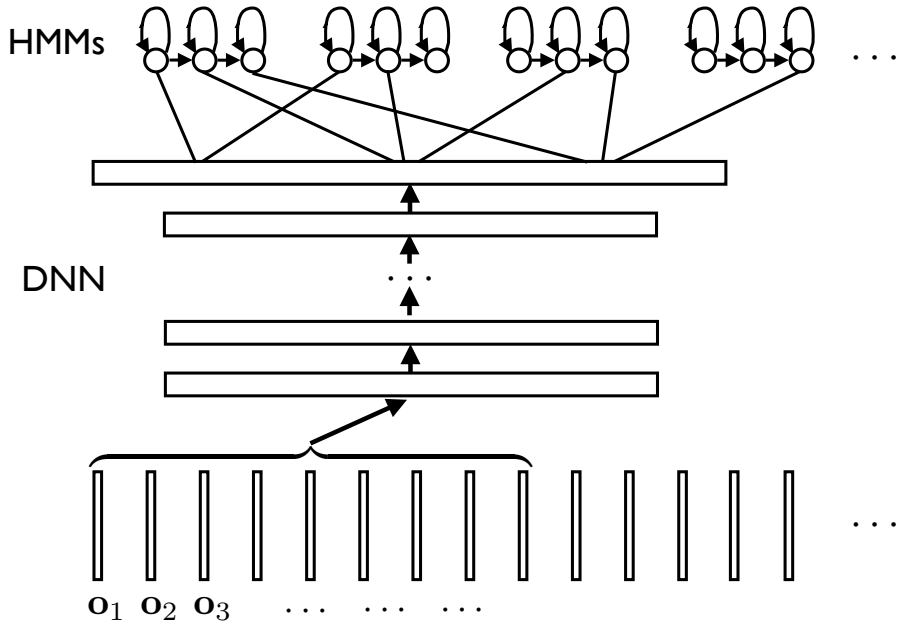

Figure 2.1.2. Context dependent deep neural network hidden Markov models (CD-DNN-HMMs)

several consecutive speech frames within a relatively long context window. In general, it is believed that the deep architecture, which employs a large amount of parameters, gives DNNs the ability to take advantage of a huge amount of training data and the non-linear hidden layers enable DNNs to model non-linearity better than GMMs. In [28] it is pointed out that the strength of DNN comes from its stronger ability to serve as a "bridge function" to alleviate the mismatch between the data distribution and the decision model

In spite of the exciting improvements, from the generative GMM to the discriminative DNN, the acoustic modeling field also suffers some losses. The CD-GMM-HMMs based acoustic model as in Figure 2.1.1 tells us the likelihood that an acoustic frame is generated by a GMM associated with an HMM state. Most of the parameters in CD-GMM-HMMs are interpretable so we can easily understand and diagnose the model. This also makes it easier for us to utilize well-established statistical theories/techniques to improve the model. However, CD-DNN-HMMs as in Figure 2.1.2 is a model discriminatively providing mapping between the acoustic frames and the HMM states. It is hard to interpret the DNN parameters and this makes the DNN more like a black box [28]. Nevertheless, the strong modeling power makes DNN a useful and practical tool and a lot

of research has been done for the DNN acoustic modeling.

In [49, 50] the second-order optimization method for DNN training called "Hessian-free" is investigated; in [51–53] divergent non-linear activation functions in hidden layers are employed. More robust frame-level objective functions for DNN training are investigated in [54]. Sequence-level discriminative training [55] also shows a nice ability in improving the DNN's discriminative power. Besides DNNs, various architectures of neural networks are explored. Convolutional neural networks (CNN) are used in [56, 57] for better robustness. In [58], it is reported that recurrent neural networks (RNN) with long short-term memory (LSTM) are achieving the state-of-the-art ASR results. Yet another way to implement a deep architecture is to extract bottleneck (BN) features with CD-DNN-HMMs and train CD-GMM-HMMs with those BN features [27, 59].

2.2 Adaptation for Automatic Speech Recognition

CD-DNN-HMMs suffer from the same performance degradations as CD-GMM-HMMs due to potential acoustic mismatches between training and testing conditions. A possible, and simple, solution to enhance ASR robustness is to collect a huge amount of training data in an attempt to create acoustic models covering all possible acoustic conditions in order to achieve the required degree of robustness. However, it is clear that robust speech recognition cannot be solved by simply collecting more data. In fact, it is difficult to collect sufficient data in various real-life conditions. Re-building a new ASR system from scratch would eventually lead to over-fitting due to limited data. For example, when we want to personalize speech recognition for a particular person, directly using the models trained with a large collection of people will obviously break the "same distribution" assumption. However, data collected from this particular person is usually far from enough to train a workable CD-GMM-HMMs system, let alone a CD-DNN-HMMs which usually employ many more parameters. Fortunately, in such circumstances, model

adaptation techniques, which have been demonstrated to be a valid and effective way to modify the existing acoustic model parameters to better resemble the testing data, can be utilized.

The adaptation approaches for CD-GMM-HMMs mainly fall into two families: maximum a posteriori (MAP) [5] based and maximum likelihood linear regression (MLLR) [6, 60] based. MAP adaptation in [5] assumes a conjugate prior distribution for the GMM-HMM model parameters, and the classical maximum likelihood estimation algorithms, such as the forward-backward algorithm [61] and the segmental k-means algorithm [62], are extended in the MAP formulation. MAP has shown to be promising to address data scarcity issues while providing a highly desirable asymptotic behavior as the number of adaptation sentences increases toward infinity. Yet one drawback of MAP is that the unseen phones can not be updated when the adaptation data is limited. Structural MAP [22] is then brought up to address this problem. SMAP takes advantage of the nice asymptotic property of MAP estimation for large size adaptation and the flexible parameter tying strategy in a tree for small sample adaptation. By assuming that the prior knowledge in a tree node can be used to construct prior density needed for MAP estimation of all the parameters in the successive child nodes, the SMAP algorithm nicely addresses the unseen phones problem. Following the idea of MAP, some online adaptation techniques are developed in [63, 64].

The basic idea behind MLLR based adaptation techniques is to estimate a transform for the model parameters. There are three main ways to perform MLLR: mean based MLLR [6, 60], variance based MLLR [65] and constrained MLLR (CMLLR) [6]. Constrained MLLR is also referred to as feature MLLR (fMLLR) because it uses constrained linear transforms to adapt both mean vectors and covariance matrices which is equivalent to a feature transform. Extensions of MLLR adaptation were also investigated in the literature, such as confidence based MLLR [66] and lattice based MLLR [67, 68]. As a combination of MAP and MLLR, in [69], the maximum likelihood estimation in MLLR was extended

to the maximum a posteriori estimation and superior results were obtained.

Besides the two families of adaptation methods, some adaptation techniques are specially designed to combat noisy environment, for example, vector Taylor series (VTS) based [70, 71] and probabilistic space maps (PS-MAPS) based [72] techniques.

Adaptation for CD-DNN-HMMs is more challenging than for the earlier connectionist systems because of the CD-DNN-HMMs' huge parameter set. Furthermore, the DNN parameters are adapted by every sample frame regardless of its senone class. Therefore, the posterior probabilities for the unobserved and scarcely seen senones are often pushed towards zero during adaptation. Such a phenomenon is commonly referred to as *catastrophic forgetting* [73]. In [74], a Kullback-Leibler divergence (KLD) based objective criterion was devised in order to alleviate the catastrophic forgetting problem. Regularization based methods are also adopted effectively to prevent catastrophic forgetting. In [75], L2 regularization is used to improve the generalization of the DNN model.

Besides the approaches mentioned above, various methods have been suggested to to improve CD-DNN-HMMs' adaptation. Transformation based methods are among the most popular adaptation techniques for CD-DNN-HMMs. They were originally devised for connectionist systems, which are the antecedent of CD-DNN-HMMs. The key idea is to augment the structure of ANN components by adding an affine transformation network to the input [76], hidden [77], or output layer [78]. They are typically trained while keeping the rest of the network parameters fixed. Motivations for these approaches stem from the concept that only relatively few parameters could be learned during adaptation. Therefore, it is preferable to train the entire network when the adaptation set is limited. More advanced transformation based methods are investigated in [79, 80]. In [81] factorized adaptation is performed both in joint factor analysis (JFA) style [82] and in VTS style [83, 84]. In [85–87], aiming at reducing the number of parameters to adjust during adaptation, the activation functions of the DNN are specially designed to be

adjustable and during adaptation only the parameters of activation functions are updated. Singular value decomposition is used in [88, 89] also with the purpose of controlling the number of parameters. I-vectors are used in [90] to encapsulate speaker identity information during speaker adaptation. In [91], a fast speaker adaptation method is proposed by learning a vector embedding called speaker code during adaptation.

2.3 System Combination for Automatic Speech Recognition

ASR can be regarded as a sequential pattern classification problem due to the dynamic nature of the speech signal [92]. There are a lot of approaches to building ASR systems, and those approaches can differ in the feature extraction method, classification approach, and training algorithm. They often use complementary information about speech, but the attempt to preserve one part of the information often leads to the loss of another. As a consequence, each individual ASR approach could only attain a different degree of success. System combination has thus been used to improve the ASR performance, and a large number of techniques are available in the literature. For example, recognizer output voting error reduction (ROVER) combines the outputs of multiple systems into a single network. Voting and/or confidence scores are then used to select the final output [7,93]. Some refined techniques based on word lattices try to minimize the approximated Bayes risk [8, 94]. The combination can also be accomplished by interpolating scores generated by several systems, e.g., [95–100].

The concept of combining several complementary classifiers to enhance the predictive performance of individual learners has been successfully applied in many scientific disciplines [101]. The idea of the *multiple classifier system* is to build a prediction model by designing a *committee* that effectively combines the strengths of a collection of simpler classifiers while avoiding the weaknesses of each individual base classifier [101, 102]. The success of system combination can be better appreciated by considering real problems involving a large number of target classes and noisy inputs (such as character recognition,

speech recognition, remote sensing, medical applications, etc.). Although there exist a number of classification algorithms based on different theories and methodologies, designing a single perfect classifier for a specific problem is often difficult to accomplish. In fact, the individual classifier ignores the uncertainty left by the finite amount data used to build it. Each classifier could thus attain a different degree of success, yet none of these classifiers is as good as needed for practical applications.

In many real-world scenarios, we face the need to lump together information from a set of imperfect classifiers. To fulfill this need, several methods of fusing multiple classifiers have been proposed over the years. In bagging [103] and random forests [104], for example, a committee of strong and complex models, usually trees, is built using different bootstrapped training data sets. Then the average of their predictions is taken with the goal of reducing the variance of each individual base model. In boosting methods, e.g. [105], the key idea is instead to combine several weak models to produce a powerful committee. The base estimators are sequentially learned on repeatedly modified versions of the data. Majority voting (or weighted combination) is then used to output the final prediction. In stacking [106], several base modes are built using the available data. A combined model is then built to make a final prediction employing all the base models as its input. In Bayesian model averaging (BMA) [107], the Bayes optimal classifier is approximated by sampling the hypothesis space and combining these hypotheses through Bayes' rule. Bayesian classifier combination (BCC) [108] aims to overcome the tendency of BMA to converge toward a single model. As opposed to sampling each ensemble individually, BCC samples from the space of possible ensembles, with model weighting randomly drawn from a Dirichlet distribution.

CHAPTER 3

DIRECT BAYESIAN SUPERVISED ADAPTATION ON AFFINE TRANSFORM PARAMETERS FOR DEEP NEURAL NETWORKS

3.1 Introduction

Recent success in adopting the CD-DNN-HMMs [30] has demonstrated significant performance improvements in the field of automatic speech recognition (ASR). Compared with the conventional context dependent Gaussian mixture model hidden Markov models (CD-GMM-HMMs) [14], the adoption of deep neural networks has shown huge gains in recognition accuracy in various tasks and datasets [2]. Unfortunately, the CD-DNN-HMMs, similarly to the conventional CD-GMM-HMMs [14], also suffer from a performance drop under potential mismatched conditions between training and testing. A degradation of the recognition accuracy is typically observed when channel conditions change, or when moving from a speaker-dependent (SD) to a speaker-independent (SI) environment due to inter-speaker variability [109]. A possible, and simple, solution to enhance ASR robustness is to collect a huge amount of training data in an attempt to create acoustic models to cover all possible acoustic conditions in speech. However, robust speech recognition cannot be solved simply by collecting more data, since the data-labeling process can be expensive to carry out. Moreover, it can also be quite complicated to collect the training data in many real-world applications, e.g., under-resourced languages [110]. Since the amount of data available for a specific condition is usually quite limited, re-building a new ASR from scratch would lead to over-fitting on these data.

Acoustic model adaptation has demonstrated to be a valid and effective approach to modifying the acoustic model parameters to better resemble the evaluation data, as testified by a large number of DNN adaptation techniques available in the recent literature, e.g., [74, 79, 81, 85, 90, 91, 111–116]. The key idea of any adaptation algorithm is like this:

starting from a pre-trained (e.g., speaker and/or task independent) speech recognition system, for a new user (or a group of users) to use the system for a specific task, a small amount of *adaptation data* is collected from that user. The collected data is employed to construct a speaker adaptive system for the speaker in the particular environment for that specific application. By doing so, the mismatch between training and testing can be generally reduced. However, adapting parameters in a CD-DNN-HMM is much more challenging than in earlier connectionist ASR systems due to the large number of network nodes and weights brought by the deep structure. Furthermore, the posterior probabilities for the unobserved and scarcely seen senones are pushed towards zero during adaptation, because all DNN parameters are adapted by every sample frame regardless of its senone class. The latter phenomenon is commonly referred to as catastrophic forgetting [73]. Conservative ad-hoc solutions for CD-DNN-HMMs have been proposed to address the catastrophic forgetting problem. For example, a Kullback-Leibler divergence (KLD) based objective criterion was devised in [74]. A variation of the standard method of assigning the target values, referred to as conservative training, was discussed in [77].

Besides the over-fitting and catastrophic forgetting problems, personalisation of a neural model creates a storage problem as well, since a different set of neural parameters has to be stored for each specific target speaker. A good adaptation technique should (i) minimize the storage requirements, since a different set of adaptation parameters needs to be stored for each different condition, (ii) combat catastrophic forgetting issues, and (iii) allow a meaningful system performance improvement even when a small amount of adaptation data is available. To meet these challenging requirements, we present a Bayesian adaptation framework to obtain maximum *a posteriori* (MAP) estimation of a small subset of parameters in CD-DNN-HMM based ASR systems. When applied to speaker adaptation, we aim at transfer learning from a well-trained deep model for a "general" usage to a "personalized" model geared towards a particular talker using a collection of speech data provided by that speaker.

In this chapter, we try to perform Bayesian adaptation directly on the discriminative DNN models. For this research direction, maximum a posteriori estimation is employed in the manner of regularization in the DNN updating formula. For supervised batch adaptation, we build the adaptation technology based on MAP estimation of the augmented linear hidden network (LHN) parameters. Under supervised speaker adaptation, the proposed MAP adaptation scheme can provide a substantial relative word error rate (WER) reduction from an already-strong speaker independent CD-DNN-HMM baseline and can consistently outperform the conventional transformation based adaptation schemes.

Following the thinking of adding regularization in adaptation, a hierarchical adaptation technique is proposed through multi-task learning (MTL) [24] devising regularization provided by some auxiliary tasks. By adding auxiliary architecture to the original DNN and performing MTL with the auxiliary tasks, we improve the learning ability of the original DNN structure and thus enhancing the discrimination power of the DNN models.

3.2 Maximum A Posteriori Adaptation of Affine Transformation Parameters for Deep Neural Networks

In this part, we attempt to perform DNN adaptation within a Bayesian framework in the spirit of maximum a posteriori (MAP) adaptation [5]. The key goal is to re-estimate parameters in an augmented linear hidden network (LHN), i.e., affine transformation, added after the last non-linear hidden layer.

3.2.1 Training of DNN

The basic structure of a deep model is shown in Figure 3.2.1. In DNNs, hidden layers are usually constructed by sigmoid units, and the output layer is a softmax layer. The values of

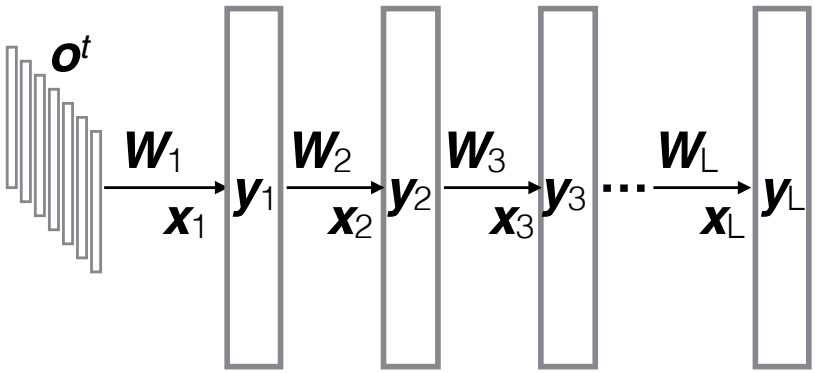

Figure 3.2.1. Structure of a deep neural network: $\mathbf{W}_i$ is weight matrix at the ith layer, note that the bias terms are omitted for simplicity.

the nodes can therefore be expressed as:

$$\mathbf{x}_i = \begin{cases} \mathbf{W}_1\mathbf{o}^t + \mathbf{b}_1, & i = 1 \\ \mathbf{W}_i\mathbf{y}_{i-1} + \mathbf{b}_i, & i > 1 \end{cases}, \tag{3.2.1}$$

$$\mathbf{y}_i = \begin{cases} \text{sigmoid}(\mathbf{x}_i), & i < L \\ \text{softmax}(\mathbf{x}_i), & i = L \end{cases}, \tag{3.2.2}$$

where $\mathbf{W}_1$, and $\mathbf{W}_i$ are the weight matrices, $\mathbf{b_1}$, and $\mathbf{b_i}$ are the bias vectors, $\mathbf{o}^t$ is the input frame at time t, L is the total number of the hidden layers, and both sigmoid and softmax functions are element-wise operations. The vector $\mathbf{x}_i$ corresponds to pre-nonlinearity activations, and $\mathbf{y}_i$ and $\mathbf{y}_L$ are the vectors of neuron outputs at the i^{th} hidden layer and the output layer, respectively. The softmax outputs were considered as an estimate of the senone posterior probability:

$$p(C_j|\mathbf{o}^t) = \mathbf{y}_L^t(j) = \frac{\exp(\mathbf{x}_L^t(j))}{\sum_i \exp(\mathbf{x}_L^t(i))}, \tag{3.2.3}$$

where C_j represents the j^{th} senone and $\mathbf{y}_L(j)$ is the j^{th} element of $\mathbf{y}_L$.

The DNN is trained by maximizing the log posterior probability over the training frames. This is equivalent to minimizing the cross-entropy objective function. Let $\mathcal{X}$ be the whole training set, which contains T frames, *i.e.* $\mathbf{o}^{1:T} \in \mathcal{X}$, then the loss with respect to

$\mathcal{X}$ is given by

$$\mathcal{L}_{xent}^{1:T} = -\sum_{t=1}^{T}\sum_{j=1}^{J} \tilde{\mathbf{p}}^t(j) \log p(C_j|\mathbf{o}^t), \tag{3.2.4}$$

where $p(C_j|\mathbf{o}^t)$ is defined in Eq. (3.2.3); $\tilde{\mathbf{p}}^t$ is the target probability of frame t. In real practices of DNN systems, the target probability $\tilde{\mathbf{p}}^t$ is often obtained by a forced alignment with an existing system resulting in only the target entry that is equal to 1, so the loss function Eq. (3.2.4) can be simplified as:

$$\mathcal{L}_{xent}^{1:T} = -\sum_{t=1}^{T} \log p(C_{tg}|\mathbf{o}^t), \tag{3.2.5}$$

where C_{tg} is the target senone at time t.

The objective function is minimized by using error backpropagation [41] which is a gradient-descent based optimization method developed for neural networks. Specifically, taking partial derivatives of the objective function with respect to the pre-nonlinearity activations of output layer $\mathbf{x}_L$, the error vector to be backpropagated to the previous hidden layers is generated:

$$\boldsymbol{\epsilon}_L^t = \frac{\partial \mathcal{L}_{xent}^{1:T}}{\partial \mathbf{x}_L} = \mathbf{y}_L^t - \tilde{\mathbf{p}}^t, \tag{3.2.6}$$

the backpropagated error vector at previous hidden layer is,

$$\boldsymbol{\epsilon}_i^t = \mathbf{W}_{i+1}^* \boldsymbol{\epsilon}_{i+1}^t \times \mathbf{y}_i \times (\mathbf{1} - \mathbf{y}_i), i < L \tag{3.2.7}$$

where $\mathbf{W}_{i+1}^*$ is the transpose of $\mathbf{W}_{i+1}$, and $\times$ denotes element-wise multiplication. With the error vectors at certain hidden layers, the gradient over the whole training set with respect to the weight matrix W_i is given by

$$\frac{\partial \mathcal{L}_{xent}^{1:T}}{\partial W_i} = \mathbf{y}_{i-1}^{1:T} (\boldsymbol{\epsilon}_i^{1:T})^*. \tag{3.2.8}$$

3.2.2 Transformation Based Adaptation for Deep Models

For DNN adaptation, some researchers choose to add an affine transformation network between the last hidden layer and the output layer weights matrix, i.e., an LHN, and adapt

only the LHN parameters while keeping all the other DNN parameters fixed [77]. In order to reduce the number of parameters to adapt, usually the last hidden layer is designed to be a bottleneck (fewer neurons).

The LIN approach performs adaptation by adding an augmented linear input layer and only adapts this set of LIN parameters. It is thought that the hidden layers of a DNN are learning a more suitable data representation for the output layer that serves as a log-linear model. By transforming the raw input using the LIN layer, we might harm the ability of data representation of the hidden layers. On the other hand for the LON approach, the issue is that usually we can't reduce the number of neurons of the output layer because we want to directly model the senones (the number of senones can be more than 10000 in practice), and that means we have to add a huge augmented layer which employ a large number of parameters.

If we deem the hidden layers as a feature extractor and the output layer as the discriminative model. The model parameters are the weights of the output layer's affine transform matrix, $\mathbf{W}_L$. The output $\mathbf{y}_L$ can now be expressed as:

$$\mathbf{y}_L = \text{softmax}(\mathbf{W}_L \mathbf{y}_{L-1}), \tag{3.2.9}$$

where the activation at the last hidden layer, $\mathbf{y}_{L-1}$, can be used as the new feature representation extracted by the hidden layers. When adding an augmented LHN after the last hidden layer, it is equivalent to applying a transformation matrix $\mathbf{W}_{lhn}$ to the model parameters to obtain an adapted model parameter set:

$$\mathbf{y}_L = \text{softmax}(\mathbf{W}_{lhn} \mathbf{W}_L \mathbf{y}_{L-1}), \tag{3.2.10}$$

An LHN adaptation structure is shown in Figure 3.2.2. This formulation is quite similar to maximum likelihood linear regression (MLLR) [60]. The difference is that in MLLR the model parameters are Gaussian mean and variance while here the model parameters are the log-linear model's transformation matrix weights.

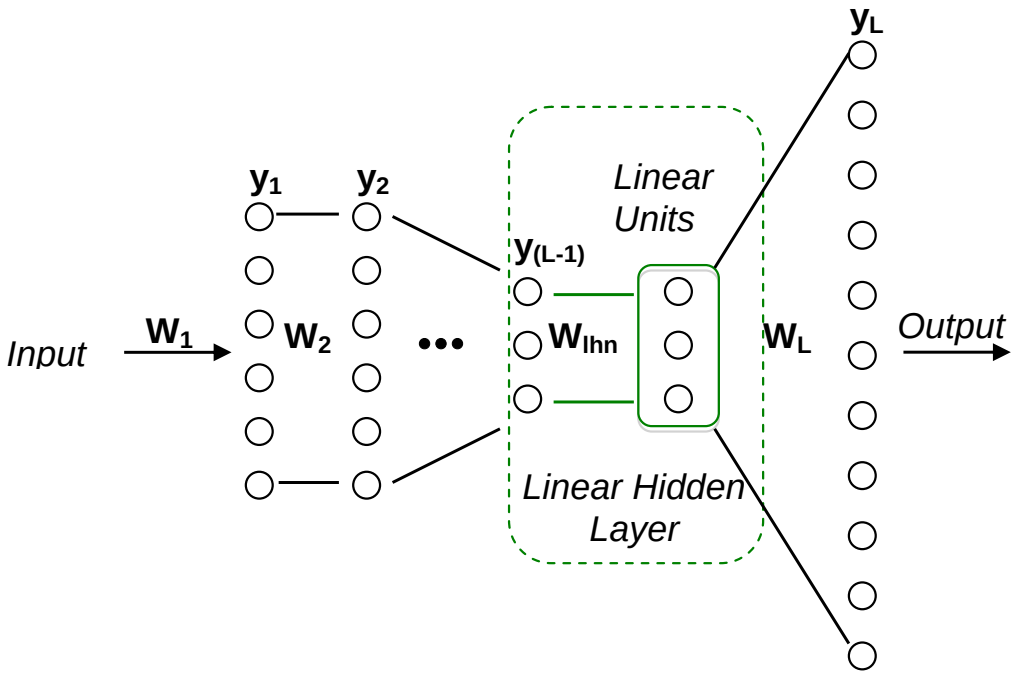

Figure 3.2.2. Basic neural architecture for adapting the HMM/ANN parameters: weights associated with the links in the dashed rectangles are estimated while all other weights remain unchanged. The activation function of each LHN units is a linear function.

3.2.3 MAP Adaptation for DNN

Although conventional DNN adaptation approaches try to alleviate over-fitting issues by reducing the number of parameters to be adapted, such number could still be very big in some cases. Inspired by the MAP adaptation that addresses the problem effectively in GMM-HMM systems, we apply the MAP approach to the LHN adaptation. Note that though we choose LHN for demonstration, the proposed MAP approach can be easily applied to other DNN adaptation frameworks like [74, 76, 78, 86, 90, 117] as well.

In order to establish a MAP adaptation framework like [5], a prior distribution over the weights of the affine transformation network needs to be imposed. To analyze and estimate the prior density, we utilized the training data of the baseline system. An empirical Bayes approach [118] is adopted. We treated each speaker in the training set as a sample speaker and performed supervised LHN adaptation those speakers. After that, we could get a particular LHN for each speaker. We observed that the histograms for weights of the adapted LHN over the training speakers are quite like Gaussian as shown in Figure 3.2.3,

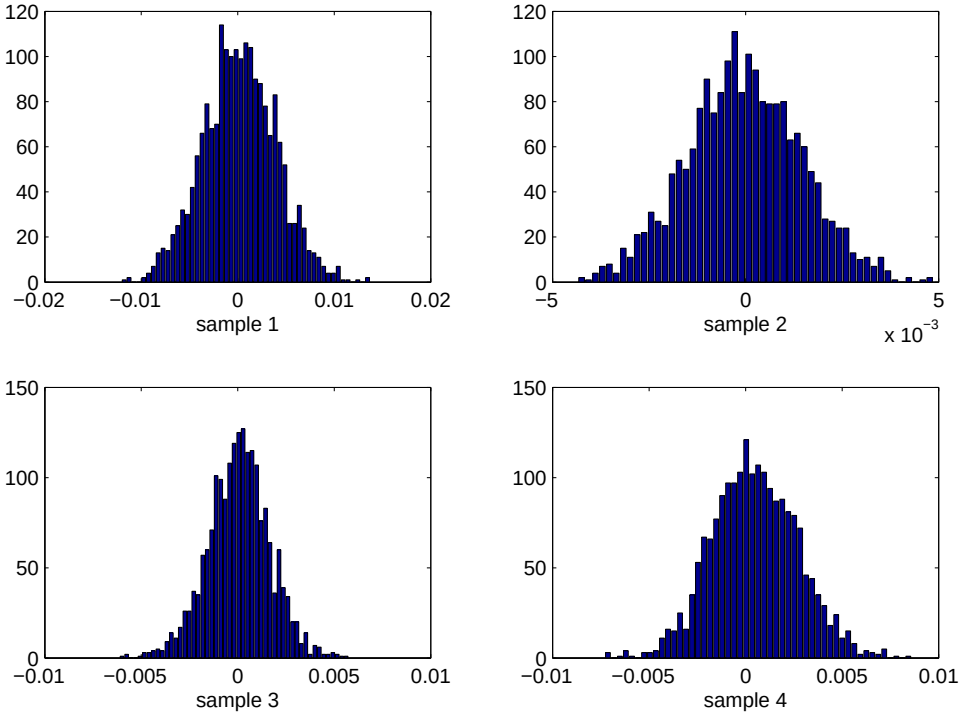

Figure 3.2.3. Histograms of 4 sample weights. The same result applies to all the weights that we do not report for the sake of saving space.

so we assume that the distribution of the weights in $\mathbf{W}_{lhn}$ to be joint Gaussian.

By expressing the weights in the LHN transformation matrix $\mathbf{W}_{lhn}$ as a vector $\mathbf{w}$ with each entry representing a particular weight, we have the prior density in the following form:

$$p(\mathbf{W}_{lhn}) = \frac{1}{(2\pi)^{M/2}|\boldsymbol{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})) \tag{3.2.11}$$

where only the diagonal entries of the covariance matrix Σ are non-zero (from the independence assumption of the weights to simplify Σ thus to reduce computational burden).

With N adapted speaker weight vectors, the maximum likelihood estimation of the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$ can be expressed as:

$$\boldsymbol{\mu}_{ML} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{w}_i \tag{3.2.12}$$

$$\boldsymbol{\Sigma}_{ML} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}_i-\boldsymbol{\mu}_{ML})(\mathbf{w}_i-\boldsymbol{\mu}_{ML})^T \tag{3.2.13}$$

where $\mathbf{w}_i$ is the vector consisting of the adapted transformation weights of speaker i.

Eq. (3.2.14) formulates the MAP learning idea by adding the term of prior density $p(\mathbf{W}_{lhn})$ to the plain cross entropy objective function.

$$\begin{aligned}\mathcal{L}_{MAP}^{1:T} &= -\sum_{t=1}^{T} \log p(C_{tg}, W_{lhn}|\mathbf{o}^t) \\ &= -\lambda \log p(W_{lhn}) - \sum_{t=1}^{T} \log p(C_{tg}|\mathbf{o}^t, W_{lhn}) \\ &= -\lambda \log p(W_{lhn}) + \mathcal{L}_{xent}^{1:T}\end{aligned} \tag{3.2.14}$$

Similar to conventional MAP learning, λ is added here to control the influence degree of the prior [119].

Applying the prior form of Eq. (4.4.4), the objective function for MAP LHN adaptation is in the form of Eq. (3.2.15).

$$\mathcal{L}_{MAP}^{1:T} = \frac{\lambda}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu}) + \mathcal{L}_{xent}^{1:T} \tag{3.2.15}$$

where only the diagonal entries of the covariance matrix $\boldsymbol{\Sigma}$ are non-zero (from the independence assumption of the weights). A close look at Eq. (3.2.15), when the prior density is a standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$, MAP learning will degenerate to conventional L2-regularized training.

The gradient of $\mathcal{L}_{MAP}^{1:T}$ with respect to $\mathbf{w}$ can now be expressed as:

$$\frac{\partial \mathcal{L}_{MAP}^{1:T}}{\partial \mathbf{w}} = \lambda(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1} + \frac{\partial \mathcal{L}_{xent}^{1:T}}{\partial \mathbf{w}}. \tag{3.2.16}$$

3.2.4 Experiments

3.2.4.1 Experimental Setup

This study is concerned with the problem of supervised batch *speaker adaptation*, and experiments are reported on the 20k-word open vocabulary Wall Street Journal task [120] using the Kaldi toolkit [121]. The baseline CD-DNN-HMM system was trained using the WSJ0 material (SI-84). The standard adaptation set of WSJ0 (si_et_ad, 8 speakers, 40 sentences per speaker) was used to perform adaptation of the affine transformation added

to the speaker-independent DNN. The standard open vocabulary 20,000-word (20K) read NVP Sennheiser microphone (si_et_20, 8 speakers x 40 sentences) data were used for evaluation. A standard trigram language model was adopted during decoding. The ASR performance was given in terms of the word error rate (WER). The reader should use caution when comparing these results with others available in the literature because we use the "20k open" test condition (a.k.a., "60k vocabulary" test condition), in which test utterances are not excluded even if they contain words not in the language model. The majority of published results are obtained with an easier evaluation condition, namely 5k and 20k-closed conditions.

The DNN has six hidden layers. The first five hidden layers have 2048 units, whereas the last hidden layer has 216 units. The output layer has 2022 softmax units. This DNN architecture follows conventional configurations used in the speech community except for the last hidden layer, which acts as a bottleneck layer. This configuration was chosen because a too large dimension of the last non-linear hidden layer might have been harmful for LHN adaptation. The bottleneck based low rank methods have been widely used to achieve more compact DNN models with equivalent performance [88, 89, 122, 123]. The number of units equals to 216. This number is chosen to simulate a sort of three-state phone layer thereby obtaining a kind of hierarchical structure between mono-phones in the hidden layer and senones at the output layer. The input feature vector is a 23-dimension mean-normalized log-filter bank feature with up to second-order derivatives and a context window of 11 frames, forming a vector of 759-dimension (69×11) input. The DNN was trained with an initial learning rate of 0.008 using the cross-entropy objective function. It was initialized with the stacked restricted Boltzmann machines by using layer by layer generative pre-training. An initial learning rate of 0.01 was then used to train the Gaussian-Bernoulli RBM and a learning rate of 0.4 was applied to the Bernoulli-Bernoulli RBMs.

3.2.4.2 Experimental Results

The word error rate (WER) attained with different adaptation techniques are reported in Table 3.2.1. All available adaptation material was used for performing adaptation, namely 40 sentences per speaker. The term BASELINE refers to the speaker independent CD-DNN-HMM system. LIN, LIN-KLD, and MAP LIN refer to the adaptation technique based on the standard linear input network approach, the KLD [74] technique in combination with LIN, and the maximum a posteriori transformation based adaptation when a prior is defined over the LIN parameters [124], respectively. The terms LON and LON-KLD are used to denote, the direct adaptation of the output layer weights matrix with or without KLD, respectively. LHN adaptation results are also reported along with the corresponding MAP version, MAP-LHN, which is the adaptation approach proposed. LHN is initialized to an identity matrix with zero bias, which gives a starting point equivalent to the unadapted model. Supervised adaptation is then performed updating only the LHN parameters. For the sake of comparison, LHN-KLD, which denotes standard LHN combined with KLD, was also evaluated.

Indeed LIN and LHN outperforms LON, which attains the worst performance improvement. KLD always improves over affine transformation based adaptation techniques, as expected. Furthermore, the proposed MAP LHN outperforms all other techniques in the given task, and it attains the best recognition results with a relative improvement of 10.4% over the BASELINE. Finally, we would like to remark that MAP LHN compares favourably against MAP LIN, and that confirms that the introduction of the bottleneck layer is the key for a proper deployment of MAP LHN.

Table 3.2.2 shows experimental results for LHN, and MAP-LHN with different amounts of adaptation sentences, namely $\{5, 10, 20, 40\}$, in the second and third columns, respectively. These results confirm that MAP-LHN adaptation almost always improves over standard LHN.

Table 3.2.1. Comparing WERs on the 20K word open vocabulary WSJ0 task for several adaptation approaches using all 40 available adaptation utterances.

System	WER (in %)
BASELINE	8.84%
LIN	8.22%
LIN-KLD	8.06%
MAP-LIN	8.13%
LON	8.80%
LON-KLD	8.64%
LHN	8.22%
LHN-KLD	8.15%
MAP-LHN	**7.92**%

Table 3.2.2. Comparing LHN and MAP LHN performance with different amounts of adaptation data.

# Adaptation Sentences	Standard LHN	MAP LHN
BASELINE	8.84%	
5	8.59%	8.54%
10	8.52%	8.52%
20	8.31%	8.12%
40	8.22%	7.92%

3.2.4.3 Hierarchical Priors: Preliminary Experiments

Hierarchical structures, such as trees, have long been used in the speech community to address the over-fitting issues during model parameters estimation. For instance, efficient adaptation with a limited amount of adaptation data was obtained through the use of a tree data structure to cluster model parameters of a CD-GMM-HMM system in [22]. Similar ideas have been recently explored in DNN learning for enhancing classification performance for classes with few examples in [125], where hierarchical priors are devised for the output layer weights matrix using a tree data structure either fixed or learnable during training.

Top-level DNN weights in a hybrid acoustic model can be regarded as senone embeddings [126], and hierarchical priors can be defined by organizing those embedding in a tree data structure. Let $\mathbf{W}_{(D+1)\times L}$ denotes the output layer weights matrix (including the bias terms). Each line in $\mathbf{W}_{(D+1)\times L}$ corresponds to a senone embedding. Specifically,

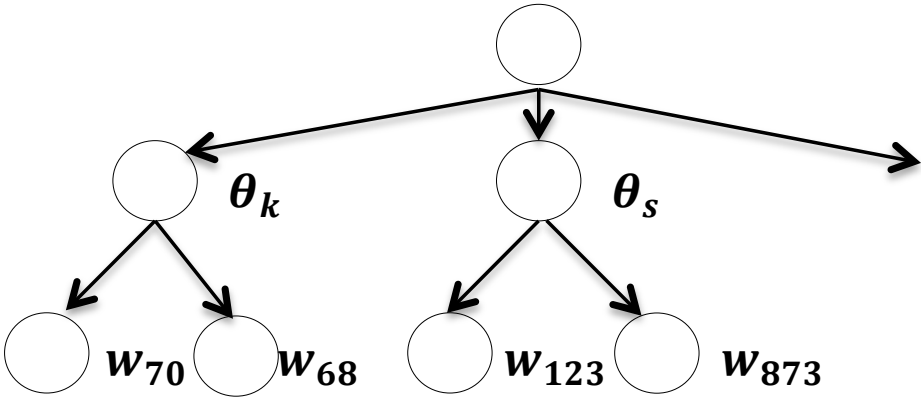

Figure 3.2.4. Fixed two-layer tree for hierarchical priors generation. Each leaf node represents a senone embedding, and it is thereby a row of $\mathbf{W}_{(D+1)\times L}$.

Table 3.2.3. WER comparisons of flat and hierarchical priors for MAP LHN with a small amount of adaptation utterances.

# Adaptation Sentences	MAP LHN Flat Priors	MAP LHN Hierarchical Priors
5	8.54%	8.48%
10	8.52%	8.45%

the sth senone embedding can be denoted as $\mathbf{w}_s$, which is the sth row in $\mathbf{W}_{(D+1)\times L}$. The tree structure used to generate hierarchical priors can be either learnt during training or given. Here, we used a fixed two-layer tree shown in Figure 3.2.4: there are L leaf nodes, with each leaf corresponding to a senone embedding; S parent nodes cluster together similar leaf nodes. Hierarchical priors can now be established by associating a vector $\mathbf{w}_s$ to a leaf node, and a vector $\boldsymbol{\theta}_s$ to each parent node, and imposing a Gaussian probability density distribution over these two vectors as follows: $\boldsymbol{\theta}_s \sim \mathcal{N}(\mathbf{0}, \frac{1}{\sigma_1^2}\mathbf{I}_{(D+1)})$, and $\mathbf{w}_s \sim \mathcal{N}(\boldsymbol{\theta}_s, \frac{1}{\sigma_2^2}\mathbf{I}_{(D+1)})$.

The objective function with hierarchical priors is in the form of Eq. (3.2.17).

$$\mathcal{L}_{1:N}^{MAP} = \mathcal{L}_{1:N}^{xent} + \frac{\lambda_2}{2}\sum\|\mathbf{w}_s - \boldsymbol{\theta}_s\|^2 + \frac{\lambda_1}{2}\|\boldsymbol{\theta}_s\|^2. \tag{3.2.17}$$

It is can be verified that $\boldsymbol{\theta}_s$ is a scaled average of all $\mathbf{w}_s$ associated to the sth leaf node by minimizing Eq. 3.2.17 over $\boldsymbol{\theta}_s$ with fixed DNN weights (see [125]). We focus our attention on experimental results with very small adaptation data amounts, as shown in Table 3.2.3.

With limited adaptation data, namely 5, 10 utterances, small performance improvements are observed against using flat priors when adaptation is carried out with hierarchical priors. Although the current improvement is still quite small, we believe more sophisticated trees can be adopted for better performance in future studies

3.2.5 Summary Remark

We have investigated a MAP adaptation approach for DNN adaptation. The key idea is to treat the parameters of the augmented affine transformation as random Gaussian variables and incorporate prior information obtained from the training data. Speaker adaptation results show that the proposed MAP approaches can lead to a consistent performance improvement over conventional LHN adaptation. Furthermore, MAP LHN outperforms other regularization schemes.

A first attempt to use hierarchical-based priors with a fixed two-layer tree structure was also studied, and small improvements were observed in a set of preliminary ASR experiments using a limited amount of adaptation sentences. Better results might still be hindered by the current fixed tree hierarchy structure employed in this preliminary work. Indeed, it was demonstrated that learning the tree hierarchy during training improves the classification performance [125]. Finally, from the objective function perspective, we are still relying on cross-entropy. Other forms of frame-level and sequence-level discriminative objectives [54, 55] can also be applied.

3.3 Hierarchical Adaptation of Affine Transformation Parameters for Deep Neural Networks via Multi-Task Learning

In this part, following the thinking of adding regularization in adaptation and continuing the effort of utilizing hierarchical structures in Section 3.2.4.3, a hierarchical adaptation technique is proposed through multi-task learning (MTL) [24] devising regularization provided by auxiliary tasks.

The key issue for CD-DNN-HMM adaptation is the large number of DNN parameters

employed in order to properly model the tied context-dependent triphone HMM states, sometimes referred to as senones [29]. This often implies that a huge number of output branch parameters connected to the output senone nodes need to be adjusted but only a small amount of adaptation data is available. Hence the posterior probabilities of both unseen and scarcely observed senones are be pushed towards zero during adaptation. In addition, DNN parameters are updated altogether by each training example, which makes it difficult to modify parameters only for a set of particular senones. As a result, the ability to model scarcely observed or unseen senones is not as well balanced with that for those well-observed senones, reducing the adaptation effectiveness. The authors in [111] proposed to directly modify the neural structure at the output layer. Specifically, an additional output layer mapping the original set of DNN output classes to a smaller set of phonetic classes was appended to the original senone output layer, and adaptation was carried out by backpropagation of errors from the new output layer. By appending the new output layer with broader sense output neuron unit, this approach successfully reduces the occurrences of unseen senones in the adaptation data. During recognition, the small output layer is removed, and the senone output layer is actually used.

Inspired by [111], we propose a novel approach to addressing the data sparsity issue by adding to the original DNN structure one or more auxiliary output layers modeling broad acoustic units, such as mono-phones, or senone clusters. With learning the original senone classification as the primary task and the phone/senone-cluster recognition as the secondary tasks, DNN is then adapted through multi-task learning (MTL), a machine learning scheme letting a classifier learn related tasks at the same time [24]. When the tasks are properly chosen, what is learned from one task can be usefully for the other tasks. MTL has been proposed for improving the generalization capability of classifiers, and it has been adopted in various speech related tasks, such as isolated digit recognition [127, 128], phoneme recognition [129], speech synthesis [130] and speech enhancement [131]. By adding the auxiliary architecture to the original DNN, and

performing MTL with the secondary tasks for recognizing broader sense acoustic units, we improve the learning ability of the original DNN structure, enlarging the coverage of acoustic space to better deal with the unseen senone problem. Thus the discrimination power of the adapted DNN models is enhanced. In this framework, the secondary mono-phone/senone-cluster classification tasks serve as auxiliary information sources for the primary senone classification task. We can control the influence of the auxiliary information by choosing proper weights for the secondary tasks.

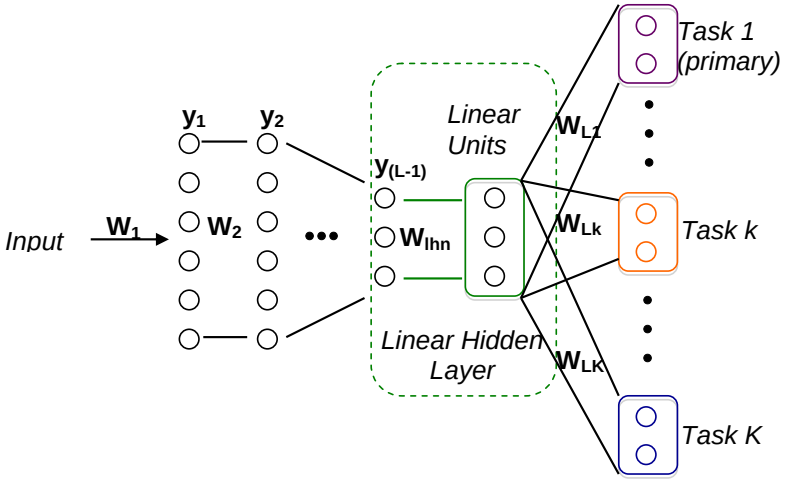

Figure 3.3.1. Basic neural architecture for adaptation of HMM/ANN models based on LHN through MTL. The output layers are associated with different tasks. In adaptation, the parameters (weights) of the LHN associated to the links in the dashed rectangle are estimated while all other weights remain unchanged. The activation function of each LHN unit is a linear function.

3.3.1 Multi-task Learning

By adding the auxiliary architecture, the new DNN will have more than one output layers. This kind of multi-output-layer DNN can be trained using the MTL scheme. MTL is a machine learning scheme letting a classifier learn more than one related tasks at a time [24]. As in Figure 3.3.1, there are multiple output layers corresponding to different tasks, and the error vector from each output layer will be backpropagated to the same hidden layer, then the error vector will be combined together in a weighted sum fashion as in

$$\boldsymbol{\epsilon}_{MTL} = \sum_{k=1}^{K} w_k \boldsymbol{\epsilon}_k \tag{3.3.1}$$

where $\boldsymbol{\epsilon}_k$ is the error vector from the kth output layer and w_k is the weight for the corresponding task. The combined error vector $\boldsymbol{\epsilon}_{MTL}$ is then backpropagated to previous hidden layers.

When the tasks are properly chosen, what is learned from one task will help the other tasks and usually there will be a primary task and several secondary tasks to aid the primary one. In this work, the primary task is classification of senones and the secondary tasks are classification of broader sense acoustic units such as mono-phones or senone-clusters. By doing so, the broader sense acoustic units in the added output layers help improve the learning ability of the original DNN structure by enlarging the coverage of acoustic space. The unseen senone problem is better dealt with and thus the discrimination power of adapted DNN models is enhanced.

The secondary tasks for limited resource adaptation need broader sense acoustic units than senones as classification targets. Mono-phones suit such purpose because a mono-phone usually corresponds to tens or hundreds of senones. To some extent, a mono-phone can be deemed as a rule-driven cluster of senones.

It is shown in [132], that a log-linear model is equivalent to a Gaussian model. The general form of a log-linear model is

$$p(C_j|\mathbf{x}) = \frac{1}{\mathrm{Z}(\mathbf{x})}\exp(\sum_i \lambda_{ji} f_i(\mathbf{x})), \tag{3.3.2}$$

where $f_i(\mathbf{x})$ is the i_{th} feature function for input $\mathbf{x}$, λ_{ji} is the weight for the j_{th} class and i_{th} feature, and $\mathrm{Z}(\mathbf{x})$ is for normalization. The mapping from a log-linear model to a Gaussian model $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ is

$$\boldsymbol{\Sigma}_j = \frac{1}{2}(\boldsymbol{\lambda}_j^2 + \Delta\boldsymbol{\lambda}^2)^{-1}, \quad \boldsymbol{\mu}_j = \boldsymbol{\Sigma}_j\boldsymbol{\lambda}_j^1 \tag{3.3.3}$$

where $\boldsymbol{\lambda}_j^2$ and $\boldsymbol{\lambda}_j^1$ are the second- and first-order weights, $\Delta\boldsymbol{\lambda}^2$ is a constant to make $\boldsymbol{\Sigma}_j$ positive definite.

The softmax function used in the DNNs' output layer can be considered as a log linear

model. From [133], we know that every senone can be represented by a multivariate Gaussian distribution with

$$\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}, \quad \boldsymbol{\mu}_j = \boldsymbol{\Sigma}[\mathbf{W}_j, \mathbf{b}_j] \tag{3.3.4}$$

where $\mathbf{W}_j, \mathbf{b}_j$ are the weights and biases for senone j and $\boldsymbol{\Sigma}$ can be an arbitrary positive definite matrix (in this work we just use an identity matrix), so for any pair of senones, the symmetric KL divergence could be used as their distance measure. With such distance measure, the large set of senones can be clustered into a small set by various clustering technologies. We use the method in [133] to generate senone clusters, which are data-driven acoustic units.

3.3.2 Linear Hidden Network Adaptation

In this work, we choose the commonly used method, linear hidden network (LHN), as the basic adaptation approach to demonstrating the effectiveness of the proposed MTL adaptation framework. The LHN adaptation is performed by adding an affine transformation network between the last hidden layer and the output layer, and adapting only the augmented LHN's parameters while keeping fixed all of the other DNN parameters [77]. In order to reduce the number of parameters to adapt, usually the last hidden layer is designed to be a bottleneck (fewer neurons). The LHN adaptation structure can be found in Figure 3.3.1. If we deem the hidden layers as a feature extractor and the output layer as the discriminative model, the LHN formulation is quite similar to maximum likelihood linear regression (MLLR) [60]. The difference is that in MLLR the model parameters are Gaussian mean and variance while here the model parameters are the log-linear model's transformation matrix weights.

3.3.3 Experiments

3.3.3.1 Baseline DNN Setup

This study is concerned with supervised *speaker adaptation*. Experiments are reported on the 20K-word open vocabulary Wall Street Journal task [120] using the Kaldi toolkit [121].

The baseline CD-DNN-HMM system was trained using the WSJ0 material (SI-84). The standard adaptation set of WSJ0 (si_et_ad, 8 speakers, 40 sentences per speaker) was used to perform adaptation of the affine transformation added to the speaker-independent DNN. The standard open vocabulary 20,000-word (20K) read NVP Senneheiser microphone (si_et_20, 8 speakers x 40 sentences) data were used for evaluation. Training was stopped using an held-out set comprising the si_dt_20 WSJ0 data. A standard trigram language model was adopted during decoding. The ASR system performance was given in terms of the word error rate (WER).

The DNN has six hidden layers. The first five hidden layers have 2048 units, whereas the last hidden layer has 216 units. The output layer has 2022 softmax units corresponding to the senones generated using a CD-GMM-HMM baseline. This DNN architecture follows conventional configurations used in the speech community except for the last hidden layer, which acts as a bottleneck layer. This configuration was chosen, because a too large dimension of the last non-linear hidden layer might have been harmful to LHN adaptation. The bottleneck based low rank methods have been widely used to achieve more compact DNN models with equivalent performance [88, 89, 122, 123]. The number 216 was chosen to simulate a sort of three-state phone layer thereby obtaining a kind of hierarchical structure between mono-phones in the hidden layer and senones at the output layer. The input feature vector is a 23-dimension mean-normalized log-filter bank feature with up to second-order derivatives and a context window of 11 frames, forming a vector of 759-dimension (69×11) input. The DNN was trained with an initial learning rate of 0.008 using the cross-entropy objective function. It was initialized with the stacked restricted Boltzmann machines (RBMs) by using layer by layer generative pre-training.

3.3.3.2 Adaptation Setup

In order to perform MTL adaptation, the auxiliary output layers should first be prepared. For each auxiliary task, we first randomly initialized the additional output layer's affine transformation parameters and then use the training data to fine-tune them while keeping all other parameters in the DNN fixed. The fine-tuning used an initial learning rate of 0.0005 with the cross-entropy based objective function. In this 20K-word open vocabulary WSJ task, the mono-phone output layer has 42 units and so is the senone-cluster output layer.

After we obtained the auxiliary output layers, an LHN was inserted between the last hidden (bottleneck) layer and the output layers' affine transform matrix. The 216×216 LHN is initialized to an identity matrix with zero bias, which gave a starting point equivalent to the unadapted model. Supervised adaptation is then performed updating only the LHN parameters. In decoding, only the original senone based output layers were used while the auxiliary architecture was discarded.

3.3.3.3 Adaptation Results

Table 3.3.1 shows the adaptation results of MTL adaptation in the 20K-word open vocabulary WSJ experiments. All the results were obtained by adding only one single auxiliary task, either mono-phone or senone-cluster classification, to the original senone classification task. We tried to include both of the two auxiliary tasks, but the results are similar to those by just adding a single task. The w in the tables means the weight of the objective function for the auxiliary task when combined with the original one. As there were only two tasks during each adaptation process, the original senone classification task's weight is therefore $1 - w$. $w = 0$ means plain LHN adaptation without the auxiliary task and $w = 1$ means there is only the auxiliary task being performed. The word "phone" means the mono-phone task and word "cluster" means the senone-cluster task.

From Table 3.3.1, it is demonstrated that the proposed MTL adaptation consistently outperforms plain LHN adaptation without MTL, especially in the case of limited

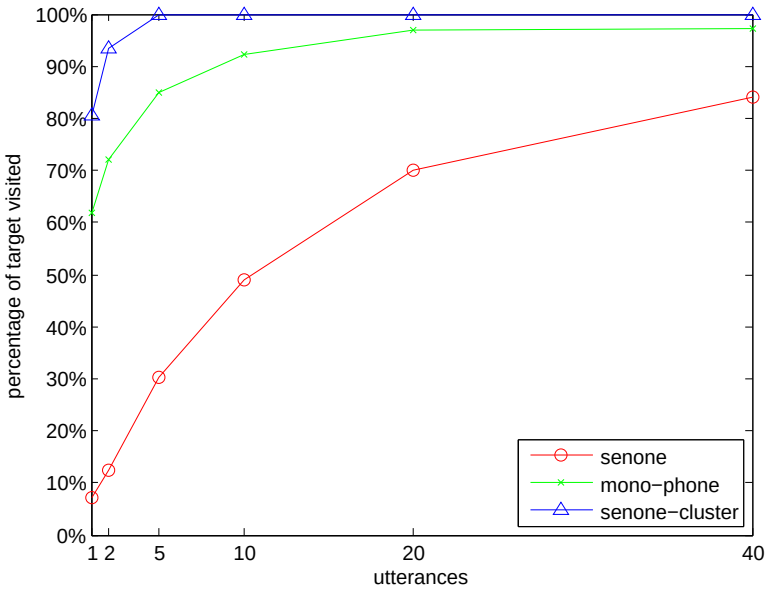

Figure 3.3.2. Percentage of the target units (senones/mono-phones/senone-clusters) visited and adapted, with respect to the number of adaptation utterances

adaptation data. 5.4% WERR is gained from speaker independent DNN with only 1 single adaptation utterance and 10.7% WERR with up to 40 utterances in the WSJ experiments.

3.3.4 Results Analysis and Discussions

Figure 3.3.2 shows the percentage of visited targets (senones/mono-phones/senone-clusters) with respect to the number of adaptation utterances. It can be observed that the coverage of the acoustic space reaches about 100% with only 5 utterances by using senone-cluster target units. The mono-phones units also shows a good characteristic by covering more than 90% the acoustic space with only 10 utterances. However, even with the complete adaptation set, i.e., 40 utterances, the adaptation data covers far below 90% of the targets when the senone units are used.

From Table 3.3.1, the best results for the extreme resource-limited cases, i.e., 1 or 2 utterances, is obtained by using senone-clusters as the auxiliary task's classification targets. This phenomenon is consistent with what is shown in Figure 3.3.2, i.e., a high coverage (80%) of the acoustic space can be reached with only 2 utterances by using senone-clusters. When the adaptation utterances increase, the coverage gap between

Table 3.3.1. WER obtained by MTL adaptation through mono-phone/senone-cluster auxiliary tasks with different weights and different amounts of adaptation data in the 20K-word open vocabulary WSJ experiments

# Adaptation Sentences	$w = 0$ (no MTL)	$w = 0.75$		$w = 1$	
		phone	cluster	phone	cluster
BASELINE	8.84%				
1	8.79%	8.56%	**8.36**%	8.65%	8.58%
2	8.58%	8.42%	**8.42**%	8.54%	8.52%
5	8.59%	**8.38**%	8.68%	8.44%	8.67%
10	8.52%	**8.33**%	8.47%	8.45%	8.42%
20	8.31%	**8.01**%	8.22%	8.45%	8.47%
40	8.22%	**7.89**%	8.17%	8.28%	8.58%

mono-phone and senone-cluster decreases, and mono-phone MTL becomes slightly better possibly due to good linguistic knowledge in contrast with the data-driven senone-cluster.

One important point needed to be mentioned is that in decoding only the original senone based output layer is used. The auxiliary architecture was discarded at this stage. But there is an interesting phenomenon that even we adapted the DNN only using the auxiliary task associated with mono-phone/senone-cluster ($w = 1$ case in the tables), there is still an improvement from the baseline DNN, and sometimes even better than only using the original primary task in some data-limited cases. This phenomenon further demonstrates that the information introduced by the auxiliary tasks can be quite effective in the limited resource scenarios.

3.3.5 Summary Remark

We propose a novel approach to addressing the data sparsity problem in CD-DNN-HMM adaptation by adding one or more small auxiliary output layers modeling broad acoustic units, such as mono-phones or senone-clusters. The DNN parameters are then updated through MTL. By doing so, we improve the learning ability of the original DNN structure, enlarge the coverage of the acoustic space to better deal with the unseen senone problem, and thus enhance the discrimination power of the adapted DNN models with limited adaptation data. We show the effectiveness of the proposed framework in the 20K-word

open vocabulary WSJ task. Experimental results show the proposed method consistently outperforms the conventional linear hidden layer adaptation scheme without MTL. With only 1 single adaptation utterance, a relative WER reduction of 5.4% is obtained from the speaker independent DNN models and a 10.7% relative WER reduction can be achieved by using 40 utterances.

A combination of the Bayesian adaptation framework in Section 3.2 and the proposed MTL scheme can be further exploited. Other kinds of auxiliary tasks such as speech enhancement and speaker verification could be also investigated. Another interesting direction is to automatically choose the sizes of the secondary output layers and the weights to combine primary and secondary tasks according to the data amount, task characteristics and other side information.

3.4 Summary

In this chapter, we try to perform Bayesian adaptation directly on the discriminative DNN models. For this direction, maximum a posteriori estimation is employed in the manner of regularization in the DNN updating formula. For supervised batch adaptation, we build the adaptation technology based on MAP estimation of the augmented linear hidden network (LHN) parameters. The proposed MAP adaptation scheme can provide a substantial relative word error rate (WER) reduction against an already-strong speaker independent CD-DNN-HMM baseline and can consistently outperform conventional transformation based adaptation schemes.

Following the idea of adding regularization in adaptation, we proposed a hierarchical adaptation technique through multi-task learning (MTL) devising regularization provided by auxiliary tasks. By adding auxiliary architecture to the original DNN and performing MTL with the secondary tasks, we improve the learning ability of the original DNN structure and thus enhancing the discrimination power of the DNN models with limited adaptation data.

CHAPTER 4

DIRECTLY BAYESIAN UNSUPERVISED BATCH AND ONLINE ADAPTATION OF ACTIVATION FUNCTION PARAMETERS FOR DEEP NEURAL NETWORKS

4.1 Introduction

Unsupervised learning is usually more realistic, and desirable, we therefore consider unsupervised adaptation in this chapter. Nonetheless, a sequential algorithm is even more attractive in real production, since it allows to adaptively track the varying parameters [10]. This learning scheme is often referred to as the *online adaptation* and makes the ASR system capable of continuously adjusting itself to a new operational environment without the requirement of storing a large set of previously used training data. The Bayesian inference theory again provides a good vehicle to formulate and solve this problem. Therefore, we also present a sequential version of the proposed MAP adaptation technique in Chapter 3, which allows us to perform unsupervised, online speaker adaptation. The implied algorithm is adaptive in nature, and it can be used to perform a full-scale online adaptive learning of the CD-DNN-HMM parameters only using the current available data to continuously track the varying acoustic conditions through the prior evolution mechanism.

In the speech recognition community, the term *unsupervised* adaptation has often been used loosely, and it actually refers to *semi-supervised* learning in machine learning, as clearly pointed out in [92]. To avoid possible confusion while assessing our experimental results, we briefly discuss here the terminology used in this chapter. Unsupervised adaptation accounts for using a seed ASR engine to decode un-transcribed data for a specific test condition, e.g. target speaker. New acoustic models, which should better resemble the test condition, are then built using these automatic transcripts as the label during acoustic model adaptation. Furthermore, the adaptation algorithm can be carried

out in a *batch*, or *online incremental* fashion. In the first case, all adaptation data are available at the same time; whereas, spoken utterances are processed one-by-one (or in mini-batches) in the latter case. It is also a common practice in the speech community to adapt and test on the same data, e.g., [112]. In this chapter, we therefore use the same NIST 2000 Hub5 evaluation material for adaptation and evaluation in order to make our experimental environment comparable to other investigations. Moreover, we analyze the *self-adaptation* performance of the proposed online MAP adaptation approach. Self-adaptation is similar to the *self-training* concept in machine learning [134], namely: the adaptation algorithm iteratively adapts a seed classifier by making predictions on the unlabeled test data, which are processed one-by-one, to expand the adaptation set [92].

We will assess the feasibility of the proposed solution on a *speaker adaptation* task and demonstrate consistent recognition error reductions on the Switchboard spontaneous speech recognition benchmark [135]. We will show through a series of comparative experiments that (i) the proposed solution is effective even when applied to already high-accuracy CD-DNN-HMMs trained in a sequence discriminative manner [55, 136], (ii) the proposed solution compares favourably against conventional linear network based adaptation schemes, and with other techniques evaluated on the same speech tasks. We also demonstrate its complementarity to other approaches by applying MAP adaptation to CD-DNN-HMM trained with speaker adaptive features, which is generated through constrained maximum likelihood linear regression (fMLLR).

To make the technology applicable to practical situations, we apply the MAP adaptation framework in Section 3.2 [137] on unsupervised online adaptation task. To deal with the data scarcity problem which becomes more severe in online adaptation, we improve upon what is proposed in Section 3.2 in several aspects. First, speaker adaptation is confined to a special linear hidden layer injected right before the softmax layer in Section 3.2; whereas, we propose to parametrize the activation function in this work. These learnable activation functions are adjusted during the adaptation phase, and this solution offers two advantages:

(i) The number of learnable parameters is limited to only twice the number of hidden nodes, and that minimises the storage requirements without introducing any special constraints that keep the number of parameters within reasonable limits. In fact, a bottleneck non-linear top layer had to be employed to constrain the amount of adaptation parameters in Section 3.2, and (ii) Adaptation equally affects all layers in the deep acoustic model rather than some specific, heuristically chosen layers, as in Section 3.2. Second, the hierarchical adaptation scheme presented in Section 3.2 is limited to the spatial dimension, and no temporal evolution of the prior distribution is investigated. In this work, we fully leverage the Bayesian framework and exploit the hierarchical relationships among prior parameters in time. That involves the time dimension through the evolution of the prior information. Third, a spontaneous speech recognition task and unsupervised adaptation is addressed in this work; whereas, a much simpler read speech task and supervised adaptation were studied in Section 3.2. Finally, self-learning is investigated in this section.

We would like to remark that feed-forward deep neural networks equipped with memory blocks [138] have been proven competitive to long short-term memory (LSTM) networks [139, 140], which represent the state-of-the-art acoustic modeling technique on the Switchboard task [141, 142]. Hence the use of the feedforward deep models is not an oversimplification with respect to the final goal of this work. Nevertheless, to further emphasise the effectiveness of the proposed technique and prove that an improvement can be demonstrated even using ASR engines attaining state-of-the-art word accuracies, we evaluate our approach against a challenging experimental scenario by employing CD-DNN-HMMs trained on speaker compensated features, which have been obtained by applying feature space transformations - referred to as fMLLR [6, 143], to the input features.

4.2 Feature/Model Space Adaptation for ASR

Broadly speaking, we can categorize speaker adaptation techniques for connectionist ASRs into two main groups, namely: feature and model spaces. In recent years, there has also been the tendency to augment the conventional speech vector with speaker-specific features that hopefully confer robustness against speaker variability. We refer to these techniques simply as *other approaches*, e.g., i-vectors [144] are employed in [145], and speaker discriminative vectors are implemented in [146]. A brief overview of the most representative adaptation techniques in each group is given in the following.

4.2.1 Feature Space Adaptation

The most representative example of feature space adaptation of deep models is the constrained maximum likelihood linear regression (MLLR) technique, referred to as CMLLR or fMLLR [6, 143]. fMLLR estimates a set of affine transforms to be applied to input acoustic features and generate speech vectors more robust to training/testing mismatches. The affine transform is found maximising the log-likelihood that the model generates the adaptation data based on first pass alignments. fMLLR has proven to be effective for adaptation of hybrid ASR engines in different tasks, e.g., [142, 147]. A major limit of fMLLR is that a CD-GMM-HMM system has to be built to generate a single input transform per each speaker. The transformed feature vectors can then be used to train the CD-DNN-HMM in a speaker adaptive manner, and another set of transforms is estimated (using the available GMMs) during evaluation for unseen speakers. The latter technique is commonly referred to as speaker adaptive training (SAT).

To simulate fMLLR without the burden of building CD-GMM-HMM systems, a linear transformation network can be added to the input of the DNN, which can be directly estimated by minimising the error at the output of the neural architecture while keeping all other DNN parameters frozen. Such a transformation rotates the input space to reduce the discrepancy between testing and training conditions. This approach is commonly referred to as linear input network (LIN) and its goal is to map the speaker dependent (SD) input

vectors to the speaker independent (SI) ASR system [76]. LIN, and its feature-space discriminative linear regression (fDLR) variant have been tested with success, e.g., [78–80].

4.2.2 Model Space Adaptation

The simplest approach to adapting a hybrid ASR system is to modify all parameters of the connectionist block using some adaptation data. Unfortunately, this solution easily leads to over-fitting on the adaptation material when the amount of data is limited [76]. A successful regularization based method to address over-fitting issues has been proposed in [74], where the Kullback-Liebler divergence (KLD) between the speaker-independent output distribution and the speaker-adapted output distributions was used during adaptation. In [88, 89, 123], a factorisation technique based on singular value decomposition was instead devised to reduce the number of parameters to be adapted. In [77], the authors proposed to add a linear transformation network before the output layer, referred to as linear hidden network (LHN). The rationale behind LHN is that the added linear layer generates discriminative features of the input pattern suitable for the classification performed at the output of the DNN. Over-fitting issues, can be further reduced by adapting the DNN top layer in a maximum a posteriori (MAP) fashion as in Section 3.2.

In [88], it was argued that the large number of DNN parameters for ASR makes adaptation challenging, and it also limits the use of environmental personalization due to the huge storage cost in large-scale deployments, and it may require a computationally intensive adaptation process. In [85], the authors proposed an ingenious solution to perform speaker adaptation for hybrid ASR systems that can simultaneously allow us to (i) reduce the computational requirements, (ii) address overfitting issues, and (iii) store the adapted parameters in a small-sized storage space. The key idea was to adapt the shape of the hidden activation functions rather than some network parameters. To this end, Hermite polynomial activation functions were used in the hidden neurons. Later, it was

demonstrated that slope and bias parameters introduced in the sigmoid activation function can be also learned in a speaker adaptive fashion [148]. Following the same line of research, an adaptive linear factor associated with each hidden unit is used to scale the unit output value and create a speaker dependent model was proposed in [149]. In practice, DNN adaptation becomes a re-weighting of the importance of different hidden units for every speaker. In the learning hidden unit contributions (LHUC) technique [112], an additional amplitude parameter is added for each hidden unit. These amplitude parameters are tied for each speaker, and are learned using unsupervised adaptation. In [80], only output layer biases have been adapted.

In the proposed approach, we assume that the activation function parameters are distributed according to a prior distribution that summarise all knowledge learned to address the source task. This prior allows us to find the most probable model with respect to the target data under MAP, which strengths robustness to data scarcity, as demonstrated in [10].

4.2.3 Other Approaches

It has been shown that robustness to speaker variability can be gained by appending speaker-specific features, computed for each speaker at both training and test stages, to the conventional speech vectors. In particular, i-vectors [144], which can be regarded as basis vectors of a speaker variability subspace, have been tested, e.g., [90, 150, 151]. Miao *et al.* [145] use an auxiliary DNN to build speaker-specific transforms of the original feature vectors.

Speaker discriminative codes, that capture speaker variabilities in trainable vectors to be used in addition to the conventional feature vectors for DNN, have been proposed in [146]. In practice, the speaker code vector is connected to a large speaker-independent neural network through a separate set of connection weights. These new weights and codes for all speakers in the training set can be jointly learned based on the available training data. Speaker codes often require speaker adaptive (re-)training, owing to the additional

connection weights between codes and the hidden units.

4.3 Connectionist ASR with Deep Models

In CD-DNN-HMM systems, the deep model estimates the *a posteriori probability*, $P(q_t|\mathbf{o_t})$, of the q_t state given the speech observation $\mathbf{o_t}$. Next, the Bayes' rule is used to obtain the observation probability, $p(\mathbf{o}_t|q_t) = P(q_t|\mathbf{o}_t)p(\mathbf{o}_t)/P(q_t)$, where $P(q_t)$ is the prior probability of each state estimated from the training set, and $p(\mathbf{o}_t)$ is independent of the word sequence and thus can be ignored.

The input to the deep architecture is typically a splice of a central frame (whose label is that for the splice) and its n context frames on both left and right sides. The hidden non-linear layers are constructed by sigmoid units, and the output layer is a softmax layer. The softmax output predicts the posterior probabilities of thousands of senones. In this work, an $(L+1)$-layer DNN, consisting of L hidden nonlinear layers ($l = 1, ..., L$) and one output layer ($l = L+1$), is used to model the posterior probability of an HMM state given an observation vector. Thus the output at the l-th hidden layer, $\mathbf{h}^l$, can be recursively defined as the nonlinear transformation of the $(l-1)$-th layer, namely:

$$\mathbf{x}^l = \sigma(\mathbf{x}^l) = \sigma(\mathbf{W}^l\mathbf{h}^{(l-1)} + \mathbf{b}^l) \tag{4.3.1}$$

where $\mathbf{W}^l$ and $\mathbf{b}^l$ are the weight matrix, and the bias vector for layer l, respectively. $\sigma(\mathbf{x}^l) = 1/(1+e^{-\mathbf{x}^l})$ is an element-wise operation. Moreover, $\mathbf{h}^l$, and $\mathbf{x}^l$ correspond to the activation and excitation of the l-th layer, respectively. Finally, $\mathbf{h}^0$ corresponds to the input observation vector.

DNN can be trained by maximizing the log posterior probability over the training frames. This is equivalent to minimizing the cross-entropy (CE) objective function. Let $\mathcal{X}$ be the whole training set, which contains T frames, *i.e.*, $\{\mathbf{o}_1, \ldots, \mathbf{o}_T\} \in \mathcal{X}$, then the loss

with respect to $\mathcal{X}$ is given by

$$\mathcal{L}^{CE} = -\sum_{t=1}^{T}\sum_{j=1}^{J} \tilde{P}_t(j) \log P(q_t^j|\mathbf{o}_t), \tag{4.3.2}$$

where $P(q_t^j|\mathbf{o}_t)$ is an estimate of the j^{th} HMM state (i.e., senone state) at time t, q_t^j, and $\tilde{P}_t$ is the target probability of frame t. In practice, the target probability $\tilde{P}_t$ is often obtained by a forced alignment with an existing system resulting in only the target entry that is equal to 1. The objective function is minimized by using error backpropagation, which is a gradient-descent based optimization method developed for artificial neural networks (see [152] for detail). In this work, we adopt mini-batch stochastic gradient descent with a chosen batch size of 256.

In [55, 136], it has been shown that sequence training can significantly boost ASR performance by incorporating acoustic models, lexicon and language models constraints in the objective function. In this work, we therefore adopt sequence training to estimate the DNN parameters of the speaker independent CD-DNN-HMM systems. Specifically, we follow [55], and the minimum Bayes risk, e.g., [153, 154], objective function at a state-level (sMBR) is used:

$$\mathcal{L}^{sMBR} = -\sum_{u} \frac{\sum_W p(\mathbf{O}_u|S_u)^k P(W) A(W, W_u)}{\sum_{W'} p(\mathbf{O}_u|S_u)^k P(W')}, \tag{4.3.3}$$

where $O_u = \{o_{u1}, \ldots, o_{uT_u}\}$ is the sequence of observation for the u-th utterance, W_u is the word-sequence in the reference for utterance u, $S_u = \{s_{u1}, \ldots, s_{uT_u}\}$ is the sequence of state corresponding to W_u, and k is the acoustic scaling factor. Finally, $A(W, W_u)$ is the raw number of correct state labels corresponding to the word sequence W with respect to the reference word sequence W_u.

Training deep neural networks from a set of randomly initialized parameters may result in a poor local optimum when performing error backpropagation. To cope with this, pre-training methods have been proposed for a better initialisation of the parameters, e.g., [43]. In restricted Boltzmann machine (RBM) based pre-training, the chief idea is to grow the DNN layer by layer without using the label information [43]. Each pair of layers in the

DNN is treated as an RBM and is trained using an objective criterion called contrastive divergence [43]. In this work, RBM pre-training is always performed.

4.4 Bayesian Learning of Hidden Activation Functions

The underpinning of the proposed MAP adaptation of trainable hidden activation functions is now presented. The first step to accomplish CD-DNN-HMM adaptation through the activation function is to parametrise the sigmoid with a slope, d, and a bias, c, that can be simultaneously learned using the adaptation data. The slope and bias terms are initialized to 1, and 0, respectively, and trained in a speaker adaptive fashion. It has been shown that adding a slope and bias term to each sigmoid accounts for pre-appending an affine linear layer, $\mathbf{\Phi} = \{\mathbf{D}, \mathbf{c}\}$, to each hidden nonlinear layer [148], where $\mathbf{D}$ is a diagonal matrix with the activation slopes, d, on the diagonal, and $\mathbf{c}$ is the vector of bias terms, c. Eq. 4.3.1, which represent the output of the l-th hidden layer, would therefore become:

$$\mathbf{h}^l = \sigma(\mathbf{x}^l, \mathbf{\Phi}^l) = \sigma(\mathbf{D}^l \mathbf{x}^{(l)} + \mathbf{c}^l) \tag{4.4.1}$$

4.4.1 Activation Function Learning

The error backpropagation algorithm can be employed to learn the aforementioned $\mathbf{\Phi} = \{\mathbf{D}, \mathbf{c}\}$ matrix while keeping all of the other CD-DNN-HMM parameters unchanged. The activation function parameters could be estimated by minimizing Eq. 3.2.4: For notational simplicity, we could expand the output vector, $\mathbf{h}^l$, in each layer by adding an additional dimension of constant 1 to incorporate the bias vector, $\mathbf{c}$, into the diagonal matrix, $\mathbf{D}$. To avoid introducing new symbols, we refer to this new weight matrix as $\mathbf{\Phi}^l$. Thus backpropagated error, $\boldsymbol{\epsilon}^l$, vector for the generating slope-bias layer is:

$$\boldsymbol{\epsilon}^l = (\mathbf{\Phi}^{l+1})^T \boldsymbol{\epsilon}^{l+1} \circ (\mathbf{h}^l)' \; (l = L, ..., 1), \tag{4.4.2}$$

where $\circ$ denotes element-wise multiplication, $(\mathbf{\Phi}^{l+1})^T$ is the transpose of $\mathbf{\Phi}^{l+1}$, and $(\mathbf{h}^l)' = \mathbf{h}^l \circ (1 - \mathbf{h}^l)$ for sigmoid. With the error vectors at certain hidden layers, the gradient over

the whole training set with respect to the weight matrix $\mathbf{\Phi}^l$ is given by

$$\frac{\partial \mathcal{L}^{CE}}{\partial \mathbf{\Phi}^l} = \boldsymbol{E}^l (\mathbf{H}^{l-1})^T. \tag{4.4.3}$$

Note that in the above equation, both $\mathbf{H}^{l-1}$ and $\boldsymbol{E}^i$ are matrices, which are formed by concatenating vectors corresponding to all the training frames from frame 1 to T, *e.g.,* $\boldsymbol{\epsilon}^l = [\boldsymbol{\epsilon}^l_1 \ldots, \boldsymbol{\epsilon}^l_t, \ldots, \boldsymbol{\epsilon}^l_T]$.

4.4.2 MAP Adaptation of Activation Functions

The maximum a posteriori (MAP) adaptation framework can be established by defining a prior distribution over the set of affine transformations, $\mathbf{\Phi}^l$s, holding the slope and bias terms for each hidden neuron.

4.4.2.1 Empirical Bayes Estimation

To analyse and estimate the prior density, we have used the training data. Moreover, we have adopted an *Empirical Bayes Approach*, e.g., [118] and treated each speaker in the training set as a sample speaker. Supervised speaker adaptive training has been performed using the data for each training speaker, separately. After that, we have obtained a set of L matrices, $\mathbf{\Phi}^l$, for each speaker. We have assumed the distribution of the $\mathbf{\Phi}^l$ matrix to be joint Gaussian.

By expressing the generic $\mathbf{\Phi}^l$ as a a vector $\mathbf{w}^l$ with each entry representing a particular slope/bias parameter, we have the prior density in a *multivariate Gaussian distribution* with the following form:

$$p(\mathbf{\Phi}^l) = \frac{1}{(2\pi)^{M/2}|\mathbf{\Sigma}^l|^{1/2}} \exp(-\frac{1}{2}(\mathbf{w}^l - \boldsymbol{\mu}^l)^T \mathbf{\Sigma}^{l^{-1}} (\mathbf{w}^l - \boldsymbol{\mu}^l)) \tag{4.4.4}$$

where only the diagonal entries of the covariance matrix $\mathbf{\Sigma}$ are non-zero (from the independence assumption). With S adapted speaker vectors, the maximum likelihood estimation of the mean $\boldsymbol{\mu}^l$ and variance $\mathbf{\Sigma}^l$ can be expressed as:

$$\boldsymbol{\mu}_{ML}^{l} = \frac{1}{S}\sum_{i=1}^{S}\mathbf{w}_{i}^{l} \tag{4.4.5}$$

$$\boldsymbol{\Sigma}_{ML}^{l} = \frac{1}{S}\sum_{i=1}^{S}(\mathbf{w}_{i}^{l} - \boldsymbol{\mu}_{ML}^{l})(\mathbf{w}_{i}^{l} - \boldsymbol{\mu}_{ML}^{l})^{T} \tag{4.4.6}$$

where $\mathbf{w}_i$ is the vector consisting of the adapted transformation weights of speaker i.

4.4.2.2 MAP Formulation

Eq. (4.4.7) formulates the MAP learning idea by adding the terms concerning prior densities, $p(\boldsymbol{\Phi}^{l})$, to the plain cross entropy objective function:

$$\mathcal{L}^{MAP} = -\sum_{l=1}^{L}\lambda^{l}\log p(\boldsymbol{\Phi}^{l}) + \mathcal{L}^{CE} \tag{4.4.7}$$

where λ^{l} controls the importance of the prior term.

Applying the prior form of Eq. (4.4.4), the objective function for MAP adaptation is in the form of Eq. (4.4.8).

$$\mathcal{L}^{MAP} = \sum_{l=1}^{L}\frac{\lambda^{l}}{2}(\mathbf{w}^{l} - \boldsymbol{\mu}^{l})^{T}\boldsymbol{\Sigma}^{l-1}(\mathbf{w}^{l} - \boldsymbol{\mu}^{l}) + \mathcal{L}^{CE} \tag{4.4.8}$$

where only the diagonal entries of the covariance matrix $\boldsymbol{\Sigma}$ are non-zero (from the independence assumption of the weights).

A close look at Eq. (3.2.15), when the prior density is a standard Gaussian $N(0|I)$, MAP learning will degenerate to conventional L2-regularized training. The gradient of $\mathcal{L}^{MAP}$ with respect to $\mathbf{w}^{l}$ can now be expressed as:

$$\frac{\partial\mathcal{L}^{MAP}}{\partial\mathbf{w}^{l}} = \lambda^{l}(\mathbf{w}^{l} - \boldsymbol{\mu}^{l})^{T}\boldsymbol{\Sigma}^{l-1} + \frac{\partial\mathcal{L}^{CE}}{\partial\mathbf{w}^{l}}, \tag{4.4.9}$$

4.4.3 Prior Evolution & Online Adaptation

The MAP adaptation method previously discussed implies batch algorithms that require processing the available adaptation data as a whole. It is often more desirable and more realistic to process the data sequentially. As discussed in [10], the advantage of a

sequential algorithm over a batch method is not necessarily in the final performance, but in computational efficiency, reduced storage requirements, and the fact that an outcome may be provided without having to wait for all the data to be processed. In addition, different data segments often correspond to different parameter values, so it is no longer desirable to process all the available adaptation samples, even if we can afford the computational load of the batch algorithm. A sequential algorithm can instead be designed to adaptively track the varying parameters, and that leads to an attractive adaptation scenario, which is known as the online (or incremental, sequential) adaptation.

Sequential methods make use of observations one at a time, or in small batches, and then discard them before the next observations are used. These methods can be used, for example, in real-time learning scenarios, where a steady stream of data is arriving, and predictions must be made before all of the data is seen. Sequential approach to learning arises naturally within the MAP adaptation framework proposed in this work, and we here present an online adaptation version based on a key concept called *prior evolution* [10]. In addition to evolution in time, priors can also be evolved in space as done in tree-based structural MAP (SMAP) [22].

In a Bayesian framework, the uncertainty of the DNN parameters is taken into account by treating these parameters, namely $\mathbf{w}^l$, as random variables. Thus our prior knowledge about $\mathbf{w}^l$, is assumed to be summarized in a known joint *a priori* pdf $p(\mathbf{w}^l|\boldsymbol{\phi}^{(0)})$ with *hyper-parameters* $\boldsymbol{\phi}^{(0)}$, where $\mathbf{w}^l \in \Omega$, and Ω denotes an admissible region of the hidden activation function parameters. The prior information can be derived from previous experience, e.g., the training data, as discussed in the previous sections. It can also be derived from previous experiences, e.g., training data, $\mathcal{X}$, as we discussed in Section 4.4.2. Now, let $\mathcal{X}^n = \{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$, be n independent sets of observation samples, which are incrementally obtained and used to update our knowledge about $p(\mathbf{w}^l)$. There exist many ways to *evolve* the prior. The central idea is that the intended evolving prior pdf $p_{intended}(\mathbf{w}^l)$ summarises the information inherited from the prior knowledge and learned

from the observation data.

Online MAP adaptation can be accomplished as follows: Given a new block of feature vector sequences, the current set of CD-DNN-HMMs is used to recognize this feature vector sequence. After the recognition of the current block of utterances, the prior pdfs for the DNN parameters, which are the results of the previous prior evolution step, are evolved to derive a set of intended posterior distributions, which will be served as the prior for the next round of prior evolution. By taking a MAP estimate from the evolved prior distributions, the hidden activation function parameters are adapted, and the updated models are used to recognize the future input utterance(s). The prior evolution algorithm requires the senone-level transcription of the speech utterances. In this work, such a transcription is derived directly from the recognition results, i.e., unsupervised adaptation.

4.4.3.1 Prior Evolution

The implementation of this learning procedure for incremental CD-GMM-HMM training raises some serious computational difficulties because of the nature of the missing data problem, and a quasi-Bayes learning formulation was proposed in [10]. In this work, we focus on prior evolution of the hidden activation parameters only, which have a multivariate Gaussian distribution, $\mathcal{N}(\cdot)$. Assuming that mean and the precision in the l-th hidden non-linear layer are unknown, then the conjugate prior for the n-th independent set is given by the normal-Wishard distribution, $\mathcal{W}(\cdot)$, [155]:

$$p(\boldsymbol{\mu}_n^l, \boldsymbol{\Lambda}_n^l) = \mathcal{N}(\boldsymbol{\mu}_n^l | \boldsymbol{\mu}_{n-1}^l, (\tau^l W_{n-1}^l)^{-1}) \mathcal{W}(\boldsymbol{\Lambda}_n^l | W_{n-1}^l, \nu) \tag{4.4.10}$$

where $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ is the precision matrix, $\nu^l < k - 1$, and W^l are called the number of degrees of freedom, and the scale matrix of the Wishart distribution (k is the dimension of the precision matrix).

The prior evolution scheme for mean and precision can now be established with ease leveraging the conjugacy properties. The mean prior evolution can be accomplished as follows (see [155] for details):

$$\boldsymbol{\mu}_n^l = \frac{\tau \boldsymbol{\mu}_{n-1}^l + |\mathcal{X}_n| \mathbf{w}^{CE}}{\tau + |\mathcal{X}_n|} \tag{4.4.11}$$

$$\tau = \tau + |\mathcal{X}_n| \tag{4.4.12}$$

where $|\mathcal{X}_n|$ is the number of observations used to perform cross-entropy based adaptation of the activation function parameters, and τ indicates the pseudo observations used to estimate $\boldsymbol{\mu}_{n-1}^l$. Eq. (3.2.12) can be used to computed $\boldsymbol{\mu}_0^l$.

The precision matrix evolution can be obtained as follows (see [155] for details):

$$W_n^l = \left((W_{n-1}^l)^{-1} + \boldsymbol{C}_{n-1}^l + \frac{\tau |\mathcal{X}_n|}{\tau + |\mathcal{X}_n|} (\mathbf{w}^{CE} - \boldsymbol{\mu}_n^l)(\mathbf{w}^{CE} - \boldsymbol{\mu}_n^l)' \right)^{-1} \tag{4.4.13}$$

$$\nu = \nu + |\mathcal{X}_n| \tag{4.4.14}$$

where $\mathbf{C}_0^l$ is initialized by multiplying Eq. 3.2.13 by S.

4.4.3.2 Online MAP Adaptation

The online MAP (OMAP) formula is similar to the batch one discussed in Section 4.4.2.2 except that the prior evolution effect has to be taken into account. The corresponding objective function is defined as:

$$\mathcal{L}_n^{OMAP} = \sum_{l=1}^{L} \frac{\lambda^l}{2} (\mathbf{w}^l - \boldsymbol{\mu}_{n-1}^l)^T (\boldsymbol{\Sigma}_{n-1}^l)^{-1} (\mathbf{w}^l - \boldsymbol{\mu}_{n-1}^l) + \mathcal{L}^{CE} \tag{4.4.15}$$

The gradient of $\mathcal{L}_n^{OMAP}$ with respect to $\mathbf{w}^l$ can be easily computed.

4.5 Experiments

4.5.1 Experimental Setup and SI Benchmarks

Two CD-DNN-HMM baseline systems were built using the 309-hour Switchboard corpus [135] - a conversational telephone speech corpus, and the Kaldi toolkit [121]. The key difference between the two baseline ASR systems relied on input speech feature

vectors used to train the connectionist component. In the first system, non-adaptive features, namely filter banks, were employed. In particular, the feature vector is a 23-dimension mean-normalized log-filter bank feature with up to second-order derivatives and a context window of 11 frames, forming a vector of 759-dimension (69×11) input. In the second system, the DNN was trained over 40-dimension adaptive feature vector, namely fMLLR, plus a context window of 11 frames, forming a vector of 440-dimension (40×11) input. As previously mentioned, fMLLR features can be generated by training a complete GMM-based system, which is then used to estimate a single input transform per speaker. The transformed feature vectors are then used to train a DNN in a speaker adaptive manner and another set of transforms is estimated (using GMM) during the testing stage for unseen speakers.

Aside from feature parametrization, the connectionist component was initialized with stacked RBMs by using layer-by-layer generative pre-training. An initial learning rate of 0.01 was used to train the Gaussian-Bernoulli RBM, and a learning rate of 0.4 was applied to the Bernoulli-Bernoulli RBMs. The DNN has six hidden layers, each having 2048 sigmoid units, with bias and slope set to zero and one, respectively, during training. The output layer has 8806 softmax units corresponding to tied-parameter context-dependent acoustic states, known as senones. Following common practices, these senone-based target units were inherited from an already-trained CD-GMM-HMM system, which had to be built to address the same task. Senone-state classes were generated using decision tree based state tying and a clustering algorithm based on the maximum likelihood criterion using the statistics collected with Gaussian models. The transition probabilities, a_{ij}, were also borrowed from the CD-GMM-HMM system [2]. The cross-entropy (CE) objective function in Eq. (3.2.4) with an initial learning rate of 0.008 was employed to start fine-tuning of the DNN parameters, which is then finalized by three iterations of sMBR sequence training. A 3-gram language model estimated from the Switchboard corpus was used in decoding.

Table 4.5.1. WERs (in %) on the SWBD data for several speaker independent (SI) CD-DNN-HMM systems using non-adaptive features. The top part of the table shows results obtained in our laboratory; whereas, WERs available in the literature on the same task and in similar experimental conditions are reported in the lower part of the table. In parentheses, the publication year is reported.

	SYSTEM	WER (in %)
IN-LAB	CE CD-DNN-HMM	16.2%
	sMBR CD-DNN-HMM	15.1%
LITTERATURE	CE CD-DNN-HMM (2011) [157]	16.1%
	CE CD-DNN-HMM (2014) [146]	16.2%
	CE CD-DNN-HMM (2016) [112]	15.2%

Experimental results are reported on the NIST 2000 Hub5 evaluation set [156], which covers 80 different speakers. The Hub5 set contains two types of data, Switchboard (SWBD) and CallHome English (CHE). SWBD data match better with the training data, since the SWBD set covers 40 speakers and 1831 utterances. Therefore, the speaker independent models generalize sufficiently well on the SWBD dataset, and the system performance improvement due to speaker adaptation is expected to be lower than that attainable in mismatched conditions. The number of spoken segments differs among test speakers, and varies between a minimum of 25 utterances to a maximum of 67 utterances per speaker (46 utterances per speaker in average). The CallHome data tend to be harder to recognize, mainly because of a greater prevalence of foreign-accented speech.

A single iteration of optimizing the MAP objective function in Eq. (4.4.9) with a fixed learning rate of 0.02 was employed during unsupervised adaptation in a batch fashion. On the other hand, unsupervised online adaptation was performed with a fixed learning rate of 0.01, and the prior was evolved as indicated in Eq. (4.4.11). The starting value of τ in Eqs. (4.4.11) and (4.4.12) was set to 1 in all experiments.

A fundamental step in our investigation on speaker adaptation of CD-DNN-HMM systems is to ensure that our baseline ASR systems are reliable. In recent years, many independent researchers worked on the NIST 2000 Hub 5 date set, yet most results are reported on the SWBD part only. To ease the comparison with what's available in the literature, we start our study by reporting the percentage of word error rate (WER) on the

Table 4.5.2. Unsupervised speaker adaptation results on the SWBD task for several techniques are reported, in terms of percentage of the WER. For easy of comparison, the first and the second column show the performance for speaker independent (SI) and speaker adapted (SA) CD-DNN-HMM systems, respectively. Non-adaptive features have been used, namely filter-bank based speech features. Consequently, WERs across these data sets are expected to be different.

SYSTEM	SI WER (in %)	SA WER (in %)
LIN	15.1%	14.9%
KLD-LIN	15.1%	14.8%
AF	15.1%	14.4%
L2-AF	15.1%	14.3%
MAP-AF	15.1%	**14.0**%
LHUC [112] (Table XII)	15.2%	14.7%
SAT-LHUC [112] (Table XII)	15.2%	14.6%

SWBD portion of the test data for different speaker independent (SI) CD-DNN-HMM systems built in our laboratory with non-adaptive features (i.e., filter-bank features) in the upper part of Table 4.5.1. By visually inspecting the results, it is easy to verify that sMBR sequence-level training indeed boosts the SI ASR performance, and error rate is meaningfully reduced from an initial 16.2% to 15.1% moving from (frame-level) CE to sMBR training. Therefore, we will use the sMBR-trained CD-DNN-HMM systems to carry out speaker adaptation.

The bottom part of Table 4.5.1 shows the best ASR performance, retrieved from the literature, attained on the SWBD dataset using non-recurrent deep models [112, 146, 157] in experimental conditions similar to ours. The comparison between the WERs obtained in our laboratory, and those available in the literature allows us to duly confirm the reliability of the results to be reported in this study. Assessing the capability of our proposed MAP adaptation techniques is our primary goal, we therefore analyze ASR performances in different adaptation scenarios in the next few sections.

4.5.2 Unsupervised Batch Adaptation: Regular Speech Features

First, we evaluate the proposed technique focusing on unsupervised batch adaptation using non-adaptive speech features (e.g., filter bank features). The SWBD dataset was used for both adaptation and testing purposes. MAP adaptation of the activation function parameters

(i.e., slope and bias terms), **MAP-AF**, which is accomplished as indicated in Eq. (4.4.9), is reported in the fifth row of Table 4.5.2. We can observe that an already high-accuracy SI sMBR CD-DNN-HMM ASR engine has been further improved from an initial WER of 15.1% down to 14.0%, indicating a relative WER reduction (WERR) of 7.3% after adaptation.

The strength of our proposed MAP scheme can be better appreciated by comparing it with other recently-published adaptation solutions. Specifically, the WER attained by applying unsupervised linear input network (LIN) speaker adaptation, **LIN**, to the SI ASR systems is reported in the first row, and second column in Table 4.5.2. In LIN, an affine transformation network is added to the input of the connectionist component. This transformation is estimated during adaptation while keeping all other DNN parameters frozen. LIN delivers a final WER of 14.9%, which corresponded to a relative WERR as small as 1.3%.

It may be argued that the KLD adaptation technique [74] could be combined with unsupervised LIN, **KLD-LIN**, to alleviate the catastrophic forgetting problem and boost the adaptation performance. KLD-LIN reduces the WER from the 15.1% down to 14.8% (see second row, and second column in Table 4.5.2), and a small improvement over plain LIN was attained, as expected[1]. This result demonstrates that the proposed approach always attains a superior performance to the conventional LIN technique yet uses a more compact representation. The number of parameters to be stored with LIN is 576,840, i.e., 23 times more parameters than those needed with the proposed solution - that uses only 24,576 parameters. The latter outcome makes the proposed approach appealing in deploying large-scale speech recognition service to the general public.

Next, the regularization capability of the proposed MAP solution can be better understood by performing unsupervised adaptation of the activation function parameters,

[1]The relative error reduction attained with KLD-based adaptation is 2%, which is in the same range of improvement reported in [74]. The latter apparently confirms the correct implementation of our KLD-based adaptation scheme.

AF, i.e. cross-entropy based training of bias and slope terms holding all remaining DNN parameters unchanged. AF can deliver a WER of 14.4%, as shown in the third row, and second column of Table 4.5.2. In the fourth row, a WER equal to 14.3% is attained in the degenerate MAP approach, that is, we introduce a L2 regularization doing unsupervised AF adaptation (please refer to Section 4.4.2.2). We can see that L2 regularization slightly enhances AF performance, but it delivers a worse recognition accuracy than the proposed MAP approach. We can therefore conclude that the ASR performance boost gained using the proposed approach is not negligible, and it is not simply due to a regularization effect.

We have already mentioned that other authors have proposed different solutions to adapt the activation function shape after Siniscalchi et al. [85]. Among these methods, LHUC [112] is the most promising. We therefore find it instructive to report the LHUC performance attained in the similar experimental conditions, and on the same ASR task in the lower part of Table 4.5.2. The WER attained with LHUC, and its improved speaker adaptive training (SAT) version, SAT+LHUC, are given in the fifth and sixth row, respectively. By comparing results shown in the fourth, fifth, and sixth rows, and first and second columns, we can confirm that the proposed MAP-AF approach outperforms both LHUC and SAT+LHUC.

4.5.3 Unsupervised Batch Adaptation: Speaker Adaptive Features

The results shown in Table 4.5.2 demonstrate the feasibility of the proposed MAP solution to the unsupervised, batch speaker adaptation problem of connectionist ASR system employing deep architectures. Nonetheless, it could be argued that baseline SI ASR engine with better seed models can be designed with either recurrent deep models, e.g., [141, 142], or feed-forward deep models equipped with memory, FFMN, [138]. In the lower part of Table 4.5.3, the experimental results reported in [138] (Table 2) of four systems utilising long-term dependency of the speech signal, namely LSTM [142], bidirectional LSTM (BLSTM) [141], and scalar feed-forward memory networks (sFFMN) [138], and vectorised feed forward memory networks (vFFMN) [138] are

Table 4.5.3. WERs (in %) w/o adaptive features on the SWBD task. The top part of the table shows results obtained in our laboratory; whereas, LHUC results in similar experimental conditions are reported in the middle part of the table. In the lower part of the table, the results with LSTM, BLST, sFFMN, and vFFMN are shown. The plus +fMLLR indicates that speaker adapted features have been used to accomplish sequential sMBR training of the connectionist component. fMLLR can be regarded as a feature-space adaptive training. The (++) symbol indicates that the following unsupervised adaptation techniques has been performed over the fMLLR-based speaker independent hybrid system.

	SYSTEM	WER
IN-LAB	sMBR CD-DNN-HMM	15.1%
	+fMLLR	13.8%
	++MAP-UF	**13.2**%
LHUC (2016) [112]	CE CD-DNN-HMM	15.2%
	+fMLLR (Table XII)	14.2%
	++LHUC (Table XII)	14.2%
	++SAT-LHUC (Table XII)	14.1%
LITTERATURE (2016)	LSTM [138]	14.2%
	BLSTM [138]	13.5%
	sFFMN [138]	14.2%
	vFFMN [138]	13.2%

Table 4.5.4. Unsupervised batch speaker adaptation results on Hub5.

	WER (in %)		
SYSTEM	**SWBD**	**CallHome**	**TOTAL**
sMBR CD-DNN-HMM	15.1%	28.7%	21.9%
+MAP-AF	14.0%	26.1%	20.1%
+fMLLR	13.8%	25.1%	19.5%
++MAP-AF	**13.2**%	**24.0**%	**18.6**%
CE CD-DNN-HMM [112]	15.2%	28.2%	21.7%
+fMLLR [112]	14.2%	26.2%	20.2%
++SAT-LHUC [112]	14.1%	25.6%	19.9%

shown. All systems were trained using filter-bank based features. WERs between 13.2% and 14.2% were attained using those systems.

An obvious questions that may arise is whether the proposed MAP-AF can further improve the state-of-the-art system performances. In this section, we provide a sound attempt to address such a challenging question. Unfortunately, we do not have access to the ASR architectures utilizing either recurrent or memory blocks, thereby we cannot directly address such a question. Nevertheless, we are mainly interested in verifying whether a performance improvement can be observed using better seed models. To this end, we

have built a speaker independent CD-DNN-HMM system using speaker adaptive features, namely fMLLR, and sequential sMBR training. The speaker compensated features allows us to reduce the WER from the initial 15.1% down to 13.8%, as indicated in the second row of Table 4.5.3. By applying MAP-AF to this new system, the WER is further brought down to 13.2%, as shown in the third row of Table 4.5.3. We can therefore conclude that MAP adaptation of the hidden activation function parameters could indeed effectively enhance state-of-the-art ASR systems. That is, our solution has the potential to boost recurrent and memory equipped deep models, since it works with an fMLLR-based ASR system that reported a recognition error between the BLSTM-based and vFFMN-based systems.

As a final remark, it may be instructive to remind that fMLLR adaptive features make the experimental setup more challenging because the proposed technique has to be applied to an already speaker adapted CD-DNN-HMM system - fMLLR can be thought of a feature-space speaker adaptation technique as seen in Section 4.2.1. On top of that, the experimental setup is further exacerbated by the intrinsic characteristics of the data: (i) the SWBD data are narrow-band, containing less information for discrimination between speakers, as discussed in [112], especially when estimating relevant statistics from small amounts of unsupervised adaptation data, and (ii) the SWBD data set exhibits a large overlap between training and test speakers - 36 out of 40 test speakers are observed in training, which allows learning more accurate speaker characteristics during supervised as opposed to unsupervised speaker adaptive training, as argued in [112]. Therefore, there is a very small room for improvement, and the fact that MAP-AF already brings the WER down to 13.2% from the initial 13.8%, which corresponds to a relative WERR equal to 4.4%, indeed confirms the viability of the proposed technique. For the sake of comparison, LHUC results are reported in the lower part of Table 4.5.3.

The set of results reported in Table 4.5.4 completes our investigation into unsupervised, batch MAP speaker adaptation. In particular, the results on the CallHome subset, and on the full 2000 Hub5 benchmark are shown. The proposed MAP speaker adaptation technique

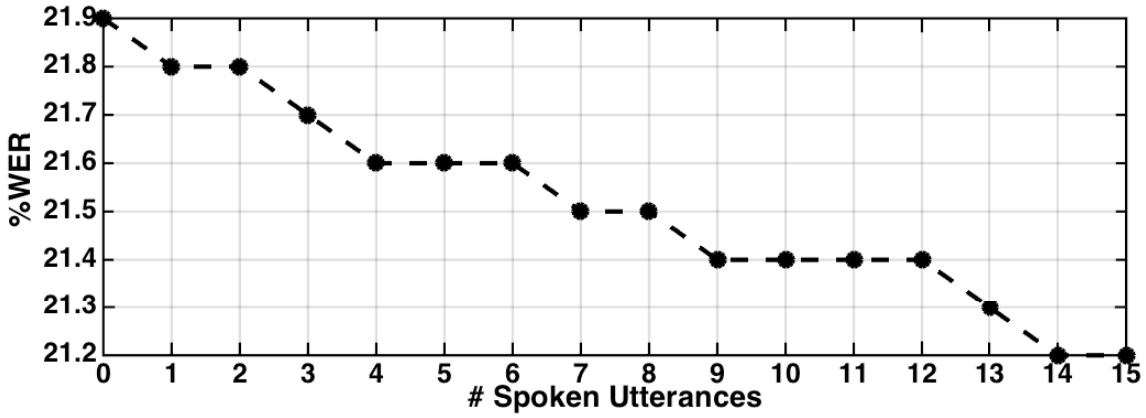

Figure 4.5.1. *Self-adaptation results, in terms of the %WER, as the number of adaptation utterances increases from 1 to 15. WERs are always given to the whole 15 utterances to make clear the effect of self-adaptation on the system performance. The SI ASR performance corresponds to that obtained with 0 adaptation utterances.*

performs relatively better under more mismatched conditions, namely the CallHome subset of the Hub5 2000 benchmark. MAP-AF also gives a consistent reduction from fMLLR, and the overall WER was reduced from 19.5% to 18.6%, as shown in the 3rd and 4th rows, respectively - a 4.6% relative WERR. In summary, we can attain up to a 15% relative WERR from the SI systems by combining fMLLR and MAP-AF.

4.5.4 Unsupervised Self-Adaptation

Next we investigate the *self-adaptation* properties of our proposed unsupervised MAP adaptation approach. To this end, we select for each speaker 15 spoken utterances for a total of 1200 utterances. The SI CD-DNN-HMM system, trained on filter bank features, attains an overall WER of 21.9% on the entire test set, as shown in Figure 4.5.1 at 0 spoken utterances. For each speaker, we perform self-adaptation using a single utterance per time, as follows: after seeing one spoken utterance, we adapt the acoustic models with MAP and test on all utterances in order to properly compare results and isolate the effect of self-adaptation. This procedure repeats until we have seen all of the 15 utterances from each speaker. In Figure 4.5.1, we display self-adaptation performance in terms of the number of adaptation utterances. As expected, adaptation consistently improves the acoustic models as the number of available adaptation utterances increases, which is a

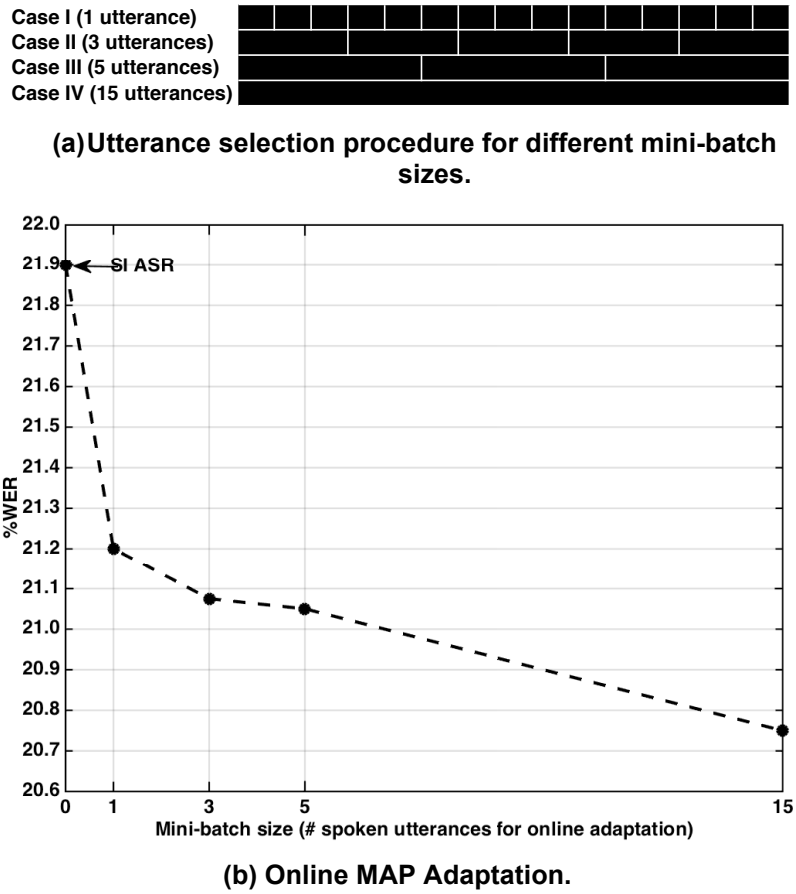

Figure 4.5.2. *Online adaptation with a varying mini-batch size. Results are averaged over all NIST 2000 Hub 5 speakers and spoken utterances.*

desirable property of any speaker adaptation scheme. Next, we can see that negative transfer learning is avoided, since a drop in the performance was avoided even with only a few adaptation sentences, e.g., 1, or 2.

4.5.5 Unsupervised Online Iterative Adaptation

Any learning framework becomes more useful for practical situations if it is performed in a sequential manner. The set of experiments reported below are meant to demonstrate that, leveraging upon the intrinsic recursive nature of Bayesian learning, online speaker adaptation can be successfully accomplished.

In real application scenarios, we can often acquire only one or a few utterances per speaker, and then the corresponding transcriptions have to be delivered to the end-user.

A sequential iterative algorithm suits such a scenario better than adaptation with a large-sized batch. A key advantage of the proposed Bayesian framework is that it is adaptive in nature and suitable for performing iterative learning. This unique adaptive characteristic is achieved by incrementally evolving the hyper-parameters, as shown in Eqs. (4.4.11) and (4.4.12), while adapting the activation function parameters. Before delving into the experimental results, we should point out that experiments with fMLLR features would require generating an affine transformation for each single iteration, and for each single speaker. That would make the experimental setup cumbersome; therefore, we report results using only non-adaptive speech feature, i.e., filter bank features.

Figure 4.5.2(b) shows that online incremental MAP adaptation brings a meaningful improvement even considering a single utterance at a time, and the WER is reduced from the initial 21.9% down to 21.2%. Next, it can be meaningful to characterize the online MAP adaptation scheme in terms of the mini-batch size, i.e., number of spoken utterances processed sequentially. Figure 4.5.2(a) shows that four different mini-batch sizes have been considered, namely {1, 3, 5, 15}; furthermore, these mini-batch have been designed so that the data are used evenly across all min-batch sizes, i.e., there is an overlap among mini-batches, and that guarantees that changes in the ASR performance are imputable solely to the mini-batch size. In Figure 4.5.2(b), we report WERs averaged over all NIST 2000 Hub 5 speakers and spoken segments for online incremental adaptation with a varying mini-batch size. Mini-batch size 0 implies no adaptation, and the SI ASR performance is given. We can observe that online MAP adaptation attains a better recognition accuracy as the mini-batch size increases, and a final WER of 20.7% is delivered with a mini-batch size of only 15 spoken utterances. The latter result confirms the viability of the proposed approach.

4.6 Summary

In this section, we have presented a theoretical framework of maximum *a posteriori* adaptation of the hidden activation function parameters in CD-DNN-HMMs. In the

experimental part of this study, adaptation is performed in an unsupervised manner, i.e., the true transcriptions of the adaptation data are assumed unknown, since that scenario is more realistic in real applications. To examine the viability of the proposed Bayesian framework, MAP adaptation is applied to a batch speaker adaptation application using the NIST 2000 Hub5 benchmark. In a series of comparative experiments, we study the the effects of different speech features, namely non-adaptive, and speaker adaptive, reporting improvements in all tested scenarios. Moreover, the effectiveness of the proposed approach was demonstrated using high-accuracy speaker independent deep models built with discriminative sequential training. The experimental results also confirmed the feasibility of the proposed techniques. Leveraging on the intrinsic recursive Bayesian nature of the proposed technique, we have also proposed an online incremental approach to unsupervised, sequential speaker adaptation by simultaneously updating the hyperparameters of the approximate posterior densities and DNN parameters on a per utterance basis. Finally, self-adaptation properties of our proposed solution have also been successfully tested. In conclusion, our experimental results indeed confirm the viability of the proposed MAP adaptation framework of hidden activation function parameters in deep models. In future studies, we intend to expand the parameter set to include other deep model parameters.

CHAPTER 5

BAYESIAN ADAPTATION ON GENERATIVE MODELS DERIVED FROM DEEP NEURAL NETWORKS

5.1 Introduction

In Chapter 3 and 4, we have presented several solutions to apply Bayesian adaptation directly on discriminative DNN. These proposed solutions apply prior information on DNN model in the form of regularization terms. The discriminative nature of the DNN models hampered our goal to replicate the nice features of the MAP/SMAP [5, 22] solutions developed for the generative CD-GMM-HMMs. The CD-GMM-HMMs based acoustic model basically gives us the likelihood that an acoustic frame is generated by a GMM associated with a HMM state. Most of the parameters in CD-GMM-HMMs are interpretable so we can easily understand and diagnose the model. This strength also makes it easier for us to utilize well-established statistical theories/techniques to improve the model. Successful examples include [5, 22]. However, CD-DNN-HMMs just provide a discriminative mapping between the acoustic frames and the HMM states, making it is hard to interpret the DNN parameters.

In our second direction towards Bayesian adaptation presented in this chapter, we go through the way that DNN is cast into a generative framework to better leverage Bayesian based technologies. We present a first step toward a Bayesian adaptation solution to utilize the prior information. Focusing on the adaptation of weights and biases in the last DNN affine transformation layer, a maximum likelihood (ML) estimation solution is devised to perform model adaptation. Next, a MAP adaptation scheme is formulated by incorporating prior information obtained by applying the proposed ML approach with the training data. Unfortunately, experimental results show that the proposed approaches are not always able to outperform conventional transformation based adaptation methods.

To regain the theoretical advantages of the MAP/SMAP adaptation solutions, we go

through an indirect way by converting the discriminative CD-DNN-HMMs to the generative CD-GMM-HMMs. Using the bottleneck (BN) features [26, 27] derived from the CD-DNN-HMMs, we were able to build deep BN-CD-GMM-HMMs (BN feature based CD-GMM-HMMs) with performance comparable to CD-DNN-HMMs. Then the Bayesian adaptation is done by applying the original MAP/SMAP technologies to the BN-CD-GMM-HMMs. Experimental results show the effectiveness of the proposed indirect Bayesian method and also demonstrate the ability of DNN in representation learning to serve as a bridge function between data distribution and the GMM based decision model [28].

5.2 Maximum Likelihood and Maximum a Posteriori Adaptation for Deep Neural Networks via Generative Casting

The discriminative nature of DNN models hampered our goal to replicate the nice features of the MAP/SMAP solutions developed for the generative CD-GMM-HMMs. We therefore present a first step toward a Bayesian adaptation solution by converting DNN to a generative model. We focus on the adaptation of weights and biases in the last DNN affine transformation layer and devise an ML estimation solution of these parameters to perform model adaptation, as in the generative GMM case. For each speaker, the adaptation material is divided into groups with same number of frames. Using each data group, one sample of adapted output layer can be obtained. These samples of output layer are used as"observations", and ML estimates are computed. Next, a MAP adaptation scheme is formulated by incorporating prior information obtained by applying the proposed ML approach with the training data. Experimental results show that the proposed approach gives better results than a more conventional transformation based approach applied to the same parameters. Unfortunately, MAP is not always able to further improve the adaptation results, yet these preliminary results are promising and support the feasibility of the proposed idea.

5.2.1 Methodology

In ML estimation, if we assume N independent and identically distributed observations, $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$, coming from a distribution with probability density function of $f(\mathbf{x}|\mathbf{w})$, where $\mathbf{w}$ is a vector of parameters, the joint density function for all observations is,

$$f(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N|\mathbf{w}) = \prod_{i=1}^{N} f(\mathbf{x}_i \mid \mathbf{w}). \tag{5.2.1}$$

If we consider $\mathbf{x}_i$ are fixed and $\mathbf{w}$ are free variable, this function will be called the likelihood,

$$\mathcal{L}(\mathbf{w}\,;\,\mathbf{X}) = f(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \mathbf{w}) = \prod_{i=1}^{N} f(\mathbf{x}_i \mid \mathbf{w}) \tag{5.2.2}$$

The value $\hat{\mathbf{w}}$ maximizing the likelihood function is the ML estimation of the parameter vector $\mathbf{w}$,

$$\hat{\mathbf{w}} = \max_{\mathbf{w}} \mathcal{L}(\mathbf{w}\,;\,\mathbf{X}) = \max_{\mathbf{w}} \prod_{i=1}^{N} f(\mathbf{x}_i \mid \mathbf{w}). \tag{5.2.3}$$

In the above formulation, observations are supposed to be generated from certain distributions, so the probability models of the distributions are naturally generative. In the speech recognition task, DNN is used as a discriminative model, it is therefore complicated to fit the DNN model into the same exact formulation.

To cast DNN into a generative framework, we divide the adaptation data into groups with same number of frames and use each data group to perform the linear output network (LON) adaptation. The updating is performed on the output layer directly. We chose to adapt the output layer, since it will ease the development of structural MAP solutions in the future. Weights associated with the last DNN affine transformation are updated while all other weights remain unchanged while performing adaptation. With each data group, LON adaptation gives us an updated LON network, which can be regarded as an observation $\mathbf{x}_i$ generated from a Gaussian distribution with mean vector $\mathbf{w}$ (which is the parameter of interest), and covariance matrix $\mathbf{\Sigma}$ (which is assumed to be fixed), we can formulate the log likelihood function as,

$$ln\mathcal{L}(\mathbf{w}\,;\,\mathbf{X}) = \alpha \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{w})^* \mathbf{\Sigma}^{-1} (\mathbf{x}_i - \mathbf{w}) \tag{5.2.4}$$

Table 5.2.1. WER obtained by the ML and MAP adaptation technologies with all available data

System	WSJ0	WSJ1
SI CD-DNN-HMM	8.84%	6.96%
LON [78]	8.80%	6.66%
LHN [77]	8.22%	6.68%
LIN [76]	8.22%	6.73%
CD-DNN-HMM+ML	8.65%	6.56%
CD-DNN-HMM+MAP	8.65%	6.53%

where α is a constant, $\mathbf{w}$ is free variable and $\mathbf{\Sigma}$ is fixed.

The ML estimator of $\mathbf{w}$ is,

$$\hat{\mathbf{w}} = \sum_{i=1}^{N} \mathbf{x}_i / N \tag{5.2.5}$$

The average of the adapted networks from the divided data groups formulates our ML estimation of the DNN adaptation.

In *maximum a posteriori* estimation, given a prior distribution $P(\mathbf{w})$ on the parameters, the most probable Bayesian estimator is,

$$P(\mathbf{w} \mid \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \frac{f(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \mid \mathbf{w}) P(\mathbf{w})}{P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)}. \tag{5.2.6}$$

Here we assume Gaussian conjugate prior distributions [155]

$$P(\mathbf{w}) = \alpha \exp(-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu}_0)^* {\mathbf{\Sigma_0}}^{-1} (\mathbf{w} - \boldsymbol{\mu}_0)), \tag{5.2.7}$$

and we can find the MAP estimator of $\mathbf{w}$ as follows

$$\hat{\mathbf{w}} = (\mathbf{\Sigma_0^{-1}} + n\mathbf{\Sigma}^{-1})^{-1} (\mathbf{\Sigma_0^{-1}} \boldsymbol{\mu}_0 + \mathbf{\Sigma}^{-1} \sum_{i=1}^{n} \mathbf{x}_i). \tag{5.2.8}$$

We can find the $\boldsymbol{\mu}_0$ and $\mathbf{\Sigma_0}$ by applying the ML adaptation technology by using the training data.

5.2.2 Experiments

This study is concerned with the problem of supervised *speaker adaptation*, and experiments are reported on the 20k-word open vocabulary Wall Street Journal task [120] using the Kaldi toolkit [121]. The reader should use caution when comparing these results

with others available in the literature, because we use the "20k open" test condition (a.k.a., "60k vocabulary" test condition), in which test utterances are not excluded even if they contain words not in the language model. The majority of published results are obtained with an easier evaluation condition, namely 5k and 20k-closed conditions. Two baseline CD-DNN-HMM systems were trained using the WSJ0 material (SI-84) and WSJ1 material (SI-284), respectively. The standard adaptation set (si_et_ad, 8 speakers, 40 sentences per speaker) was used to perform adaptation of the last hidden DNN layer affine transformation of speaker-independent DNN. The standard open vocabulary 20,000-word (20K) read NVP Senneheiser microphone (si_et_20, 8 speakers x 40 sentences) data were used for evaluation. A standard trigram language model was adopted during decoding. The ASR performance was given in terms of the word error rate (WER).

The DNN architecture has six hidden non-linear layers in all configurations. The input feature vector is a mean-normalized log-filter bank feature with up to second-order derivatives and a context window of 11 frames (FBANK). The DNN output layer has 2022 and 3462 Softmax units for the WSJ0, and WSJ1 configuration, respectively. The DNN configuration follows that conventionally adopted in the speech community except for the last hidden non-linear layer, which acts as a bottleneck layer. This configuration was chosen to generate BN features. The number of units in the bottleneck layer is 216, and it was chosen to simulate a sort of three-state phone layer thereby simulating a kind of hierarchical structure between mono-phones in the hidden layer, and senones in the output layer. All DNNs were trained with an initial learning rate of 0.008 using the cross-entropy objective function. DNN parameters were initialized with the stacked restricted Boltzmann machines by using layer by layer generative pre-training. Training is stopped using early stopping on a validation set spilt off the training material, namely 10% of the training data [121].

Experimental results on the 20k-word open vocabulary Wall Street Journal task are given in Table 5.2.1. Speaker-independent performance is shown in the first row for the

CD-DNN-HMM baseline system for easy of comparison. In the last two rows, the recognition performance for the proposed ML and MAP approaches is given, respectively. First, we can notice that the improvement over the speaker independent CD-DNN-HMM system is obtained with both approaches, which demonstrated that casting DNN adaptation within a *generative* framework is indeed useful. Next, we report adaptation results for three common affine transformation based adaptation schemes devised under the connectionist ASR framework to better appreciate the effectiveness of our solutions. In particular, LON, LHN, and LIN adaptation schemes are evaluated in the same experimental conditions, and the corresponding adaptation results attained by tuning the learning rates on the available adaptation material and using the decoding performance on the adaptation data as early stopping criterion are displayed in the third, fourth and fifth row in Table 5.2.1, respectively. From the table, we can see that while LHN and LIN can outperform the proposed approach on the simpler WSJ0 working condition, LON cannot. Moreover, none of three affine transformation based solutions can deliver a good improvement with the WSJ1 configuration, and both proposed solutions outperform LON, LHN, and LIN as the acoustic models cover a wider acoustic space.

5.2.3 Summary Remark

In this work, we present a first step toward DNN adaptation by casting DNN into a generative framework. We focus on the adaptation of weights and biases in the last DNN affine transformation layer and devise an ML estimation solution of these parameters to perform model adaptation. MAP adaptation scheme is then formulated by incorporating prior information obtained by applying the proposed ML approach with the training data. Experimental results show that the proposed approach gives better results than more conventional transformation based approach applied to the same parameters. Unfortunately, MAP is not always able to further improve the adaptation results, yet these preliminary results are promising and support the feasibility of the proposed idea.

5.3 MAP/SMAP Adaptation for GMM with BottleneckFeature Derived from DNN

To regain the theoretical advantages of the MAP/SMAP solutions, we go through an indirect way by converting the discriminative CD-DNN-HMMs to the generative CD-GMM-HMMs by using DNN derived bottleneck feature.

The impressive performance gain achieved by the CD-DNN-HMMs can be partially attributed to the use of high-dimension input features, namely a splice of a current frame along with its n context frames before and after. Unfortunately, high-dimensional vectors cannot be properly modeled by diagonal-covariance GMMs because consecutive frames are highly correlated, but bottleneck (BN) features, generated with DNNs, have been proven to be a viable solution to boost the performance of the CD-GMM-HMM systems [26, 27, 43]. It has recently been reported that a CD-GMM-HMM system built around BN features can compete with the state-of-the-art CD-DNN-HMM system [158]. A key advantage of the CD-GMM-HMM system built on top of discriminative BN features is that the well-established model adaptation techniques developed within the Bayesian framework in the past, such as MAP/SMAP [5, 22], could be utilized easily without any change.

5.3.1 Tandem System with Bottleneck Feature

Figure 5.3.1 is the diagram of a tandem system with bottleneck (BN) features [26]. To extract BN features, a DNN with a bottleneck layer is first trained. The output of the bottleneck layer goes through a dimension reduction stage, usually principle component analysis (PCA), and then is concatenated with the input feature to formulate the BN features. The BN features are used to train a CD-GMM-HMM system.

The conventional method to extract BN features is to use a small hidden layer, *bottleneck layer*, in the middle of the neural architecture. Then PCA is performed to de-correlate and reduce the dimension of the bottleneck layer's output. In [158], it was argued that there is no longer need for a bottleneck layer, and the output of the last hidden layer can be directly utilized as a DNN based speech feature vector; nonetheless, a

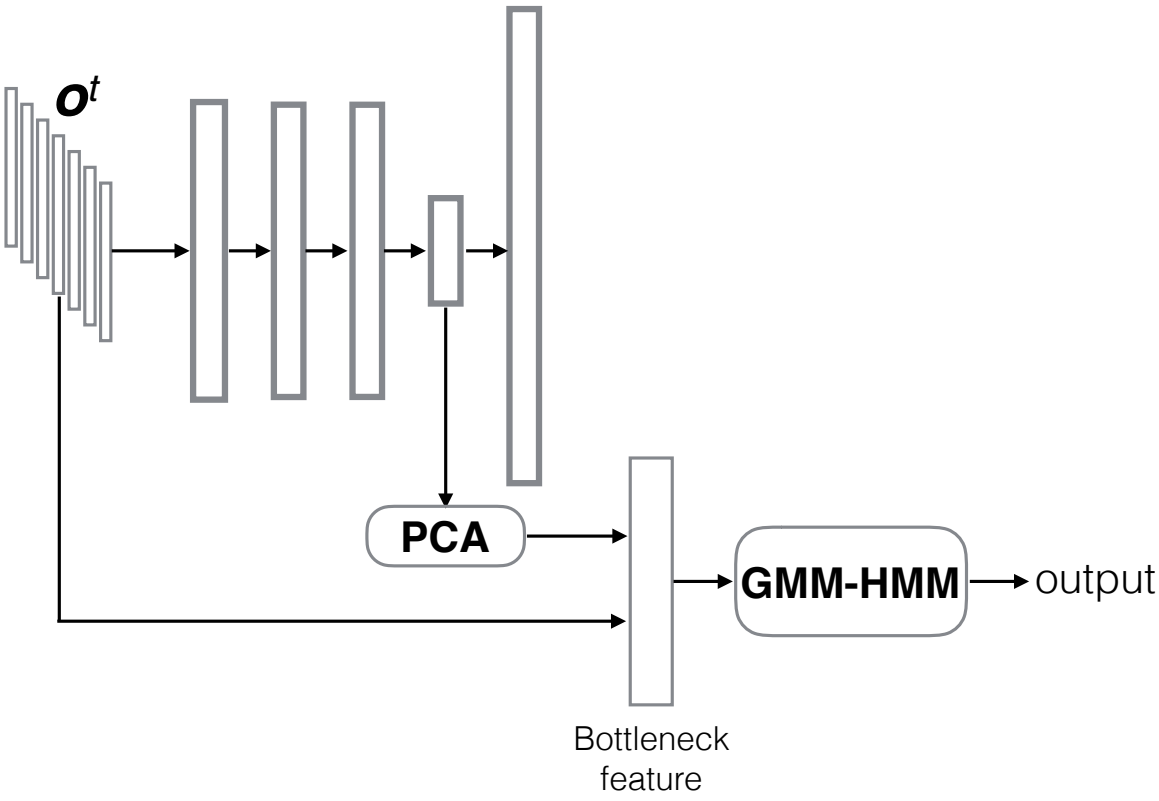

Figure 5.3.1. Tandem system with bottleneck feature

dimensionality reduction approach has to be applied to compress the dimension of these features. In this work, we decided to combine those two above solutions in order to fully utilize DNN's power of discriminative feature extraction, while avoiding performing two dimension reduction steps as in [158]. In practice, we had the last DNN hidden non-linear layer play the role of the bottleneck layer. Moreover, short-term spectral features, namely 39-dimensional MFCC+velocity+acceleration features with speaker-based cepstral mean normalization, are appended to the BN features in order to compose a new input feature vector to be used in the CD-GMM-HMM training phase. Finally, linear discriminant analysis (LDA) is applied to project the features onto a 40-dimensional space.

The CD-GMM-HMM training phase using LDA features was carried out as follows: First, a discriminative maximum likelihood linear transform (MLLT) [159] step was performed to obtain an LDA+MLLT model, and the alignments generated with a conventional CD-GMM-HMM were used to perform the MLLT transformation. Next, a

single-pass re-estimation was performed to initialize a single-Gaussian GMM for each HMM tied-state with the decision-tree state-tying structure adopted from a conventional CD-GMM-HMM system. Finally, standard maximum likelihood (ML) based CD-GMM-HMM training procedure was applied to increase the number of mixture components.

5.3.2 MAP and Structural MAP Adaptation

MAP adaptation of CD-GMM-HMMs has proven to be effective in the ASR literature. Let $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$ be a sample of T i.i.d. observations drawn from a mixture of K p-dimensional multivariate normal densities. The joint p.d.f. is specified by Eq. 5.3.1

$$f(\mathbf{x}|\mathbf{\Lambda}) = \prod_{t=1}^{T} \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x}_t|\mathbf{m}_k, \mathbf{r}_k). \tag{5.3.1}$$

Where

$$\mathbf{\Lambda} = (w_1, ..., w_K, \mathbf{m}_1, ..., \mathbf{m}_K, \mathbf{r}_1, ..., \mathbf{r}_K) \tag{5.3.2}$$

is the parameter vector. w_k denotes the mixture gain for the kth mixture component subject to the constraint $\sum_{k=1}^{K} w_k = 1$. $\mathcal{N}(\mathbf{x}_t|\mathbf{m}_k, \mathbf{r}_k)$ is is the kth normal density function denoted by

$$N(\mathbf{x}_t|\mathbf{m}_k, \mathbf{r}_k) \propto |\mathbf{r}_k|^{1/2} \exp[-\frac{1}{2}(\mathbf{x}_k - \mathbf{m}_k)^* \mathbf{r}_k (\mathbf{x}_k - \mathbf{m}_k)] \tag{5.3.3}$$

where $\mathbf{m}_k$ is the p-dimensional mean vector, and $\mathbf{r}$ is the $p \times p$ precision matrix. As stated in [5], no sufficient statistic of a fixed dimension exists for the parameter vector $\mathbf{\Lambda}$, so no joint conjugate prior density can be specified. In [5], by assuming independence assumed between the parameters of the individual mixture components and the set of the mixture weights, the approximate joint prior density $G(\mathbf{\Lambda})$ can be represented by the product of Dirichlet (g_d) and normal-Wishart (g_n) density as in 5.3.4:

$$G(\mathbf{\Lambda}) = g_d(w_1, ..., w_K) \prod_{k=1}^{K} g_n(\mathbf{m}_k, \mathbf{r}_k), \tag{5.3.4}$$

where

$$g_d(w_1, ..., w_K|v_1, ..., v_K) = \prod_{k=1}^{K} w_k^{v_k-1} \tag{5.3.5}$$

$$g_n(\mathbf{m}_k, \mathbf{r}_k|\tau_k, \boldsymbol{\mu}_k, \alpha_k, \mathbf{u}_k) = |\mathbf{r}_k|^{(\alpha_k-p)/2} \exp[-\frac{\tau_k}{2}(\mathbf{m}_k - \boldsymbol{\mu}_k)^* \mathbf{r}_k(\mathbf{m}_k - \boldsymbol{\mu}_k)] \exp[-\frac{1}{2} tr(\mathbf{u}_k \mathbf{r}_k)] \tag{5.3.6}$$

The classical maximum likelihood estimation algorithms, namely, the forward-backward algorithm and the segmental k-means algorithm, are expanded, and MAP estimation formulas are developed in [5]. Use Gaussian mean vector adaptation as example, the estimated Gaussian mean vector by MAP is given by:

$$\hat{\mathbf{m}}_k = \frac{\tau_k \boldsymbol{\mu}_k + \sum_{t=1}^{T} c_{kt} \mathbf{x}_t}{\tau_k + \sum_{t=1}^{T} c_{kt}} \tag{5.3.7}$$

where c_{kt} is the responsibility of frame $\mathbf{x}_t$ to kth mixture component.

SMAP is designed to improve the MAP estimates obtained when the amount of adaptation data is small. Taking advantage of the nice asymptotic property of MAP estimation for large size adaptation and the flexible parameter tying strategy in a tree for small sample adaptation and by assuming that the prior knowledge in a tree node can be used to construct prior density needed for MAP estimation of all the parameters in the successive child nodes, SMAP algorithm is developed in [22]. A clustering algorithm based on the Kullback-Leibler divergence between two Gaussian components, and the k-means procedure is used. We cannot delve into the specific details of the tree-based clustering procedure due to space constrains, and the interested reader is referred to [22] for more details. We assess the MAP and SMAP approaches for speaker adaptation of CD-GMM-HMMs with BN-based features demonstrating these Bayesian motivated techniques could still play a key role in speaker adaptation of deep models.

5.3.3 Experiments

This study is concerned with the problem of supervised *speaker adaptation*. using the Kaldi toolkit [121]. Two baseline CD-DNN-HMM systems were trained using the WSJ0

Table 5.3.1. MAP and SMAP adaptation results with 40 utterances against WSJ0 and WSJ1 baseline systems

System	WSJ0	WSJ1
CD-DNN-HMM	8.84%	6.96%
CD-GMM-HMM-BN	8.86%	6.91%
CD-GMM-HMM-BN + MAP	8.38%	6.89%
CD-GMM-HMM-BN + SMAP	7.85%	6.45%

material (SI-84) and WSJ1 material (SI-284), respectively. The standard adaptation set (si_et_ad, 8 speakers, 40 sentences per speaker) was used to perform adaptation of the last hidden DNN layer affine transformation of speaker-independent DNN. The standard open vocabulary 20,000-word (20K) read NVP Senneheiser microphone (si_et_20, 8 speakers x 40 sentences) data were used for evaluation. A standard trigram language model was adopted during decoding. The ASR performance was given in terms of the word error rate (WER).

The DNN architecture has six hidden non-linear layers in all configurations. The DNN output layer has 2022 and 3462 Softmax units for the WSJ0, and WSJ1 configuration, respectively. The DNN configuration follows that conventionally adopted in speech community except for the last hidden non-linear layer, which acts as a bottleneck layer. This configuration was chosen to generate BN features. The number of units in the bottleneck layer is 216, and it was chosen to simulate a sort of three-state phone layer thereby simulating a sort of hierarchical structure between mono-phones in the hidden layer, and senones at the output layer. All DNNs were trained with an initial learning rate of 0.008 using the cross-entropy objective function. DNN parameters were initialized with the stacked restricted Boltzmann machines by using layer by layer generative pre-training. An initial learning rate of 0.01 was then used to train the Gaussian-Bernoulli RBM and a learning rate of 0.4 was applied to the Bernoulli-Bernoulli RBMs.

Table 5.3.1 shows the adaptation results with all available adaptation material, namely 40 sentences per speaker, against different speaker independent baseline systems. From the table, we can see that consistent with [158], CD-GMM-HMMs with DNN derived BN

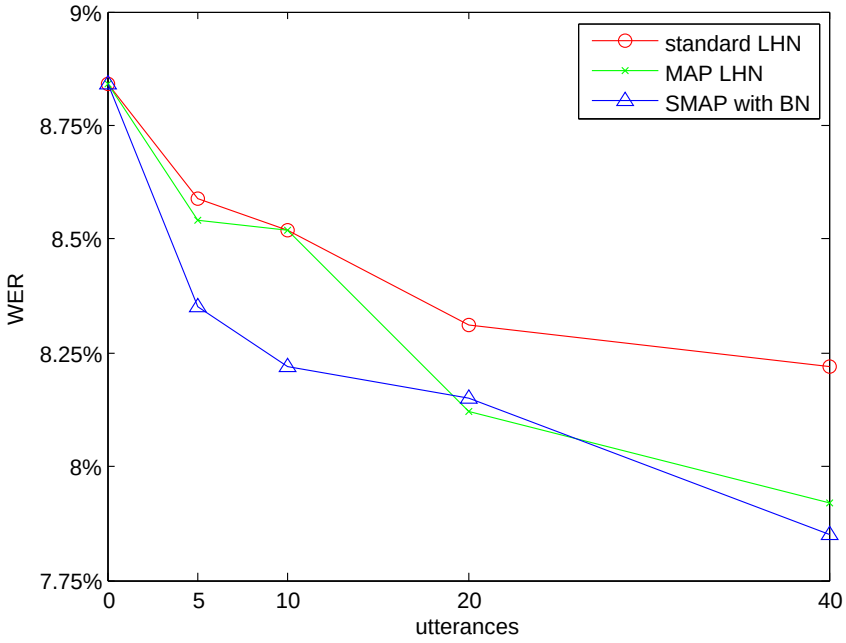

Figure 5.3.2. Comparing the adaptation results by standard LHN, MAP LHN in Section 3.2 and SMAP with BN feature, respectively

Table 5.3.2. WER obtained by SMAP with different amount of adaptation data against WSJ0 and WSJ1 baseline systems

System	WSJ0	WSJ1
CD-GMM-HMM-BN	8.86%	6.91%
+SMAP(5 utt)	8.35%	6.70%
+SMAP(10 utt)	8.22%	6.62%
+SMAP(20 utt)	8.15%	6.54%
+SMAP(40 utt)	7.85%	6.45%

feature, can achieve comparable results with CD-DNN-HMMs. For WSJ0 baseline, both MAP and SMAP can provide notable WERR. For WSJ1 baseline, MAP can not obtain significant improvement against SI system, but SMAP gives prominent WERR. WSJ1 dataset has about 80 hours of training data, while WSJ0 has only 14 hours, to some extent, the abundance of training data helps WSJ1 system enlarge the coverage of acoustic space across different speaker characteristics, so MAP can't do a good job against the WSJ1 baseline system, but SMAP, by extracting structural information from the available adaptation data, can still reduce WERR in a notable scale. This is also demonstrated in Table 5.3.2, where WERs obtained by SMAP with different amount of adaptation

sentences, namely 5, 10, 20 and 40, are reported. It can be seen that with limited adaptation dataset, SMAP can do a great job; when the adaptation dataset becomes larger, SMAP can still maintain its performance. Moreover, we compared adaptation results attained by standard linear hidden layer adaptation (LHN), MAP LHN in Section 3.2 and BN feature based SMAP proposed in this work in Figure5.3.2. From the figure, SMAP shows good performance with limited data (5, 10 utterances), and is almost steadily better than standard LHN and MAP LHN as the number of adaptation data increases.

5.3.4 Summary Remark

In this section, we go through an indirect way to perform Bayesian adaptation for deep models by converting the discriminative CD-DNN-HMMs to generative CD-GMM-HMMs. Using the bottleneck (BN) features [27] derived from the CD-DNN-HMMs, we are able to build deep BN-CD-GMM-HMMs (BN feature based CD-GMM-HMMs) with performance comparable to CD-DNN-HMMs. Then the Bayesian adaptation is done by applying the original MAP/SMAP technologies to the BN-CD-GMM-HMMs to regain the theoretical advantages of the MAP/SMAP solutions. Experimental results show the effectiveness of the proposed indirect Bayesian method and also demonstrate the ability of DNN in representation learning to serve as a bridge function between data distribution and the GMM based decision model [28].

5.4 Summary

In this chapter, we present our efforts towards Bayesian adaptation of deep models by applying adaptation on generative models derived from DNN. In our first attempt, we try to cast DNN into a generative framework to facilitate the deploy of classical ML and MAP techniques. With the proposed ML-based adaptation scheme for CD-DNN-HMMs, better results than the conventional transformation based approaches are attained. Unfortunately, MAP is not always able to further improve the adaptation results, yet these preliminary results are promising and support the feasibility of the proposed idea. In our second

attempt, we have demonstrated that MAP and SMAP adaptation are still effective in speaker adaptation with deep models. Experimental evidence demonstrates that SMAP in combination with BN-based features can deliver a meaningful recognition accuracy improvement. It supports our initial intuition that Bayesian motivated adaptation techniques might work better with generative models because of the mathematical attraction and the large sample properties.

CHAPTER 6

BAYESIAN SYSTEM COMBINATION FOR ADAPTATION OF DEEP MODELS

6.1 Introduction

The discriminative CD-DNN-HMMs adapted with MAP-LHN in the direct manner and the generative BN-CD-GMM-HMMs adapted with SMAP in the indirect manner utilize different model assumptions and are adapted with highly different techniques. As a result, the potential of them to have complementarity is promising. A novel system combination scheme is proposed to leverage the complementarity for improving the adaptation performance. The proposed scheme performs the system combination in a dynamic way exploiting the temporal nature of HMMs thus supports real-time multi-stream speech recognition.

The concept of combining several complementary classifiers to enhance the predictive performance of individual learners has been successfully applied in many scientific disciplines [101]. The idea of a *multiple classifier system* is to build a prediction model by designing a *committee* that effectively combines the strengths of a collection of simpler classifiers while avoiding the weaknesses of each individual base classifier [101, 102]. The necessity of system combination can be better understood by considering real problems involving a large number of target classes, and noisy inputs, such as character recognition, speech recognition, remote sensing, medical applications, etc. Although there exist a number of classification algorithms based on different theories and methodologies that could be used to address those tasks, designing a single perfect classifier for a specific problem is often difficult to achieve. In fact, the individual classifier ignores the uncertainty left by the finite amount data used to build it. Each classifier could thus attain a different degree of success, yet none of these classifiers is as good as needed for practical applications.

In many real-world scenarios, we therefore need to lump together information from a set of imperfect classifiers, and several methods of fusing multiple classifiers have been proposed over the years. In bagging [103] and random forests [104], for example, a committee of strong and complex models, usually trees, is built using different bootstrapped training data sets. Then the average of their predictions is taken with the goal to reduce the variance of each individual base model. In boosting methods, e.g., [105], the key idea is instead to combine several weak models to produce a powerful committee. The base estimators are sequentially learned on repeatedly modified versions of the data. Majority voting (or weighted combination) is then used to output the final prediction. In stacking [106], several base modes are built using the available data. A combined model is then built to make a final prediction employing all the base models as its input. In Bayesian model averaging (BMA) [107], the Bayes optimal classifier is approximated by sampling the hypothesis space and combining these hypotheses through Bayes' rule. Bayesian classifier combination (BCC) [108] aims to overcome the tendency of BMA to converge toward a single model. As opposed to sampling each ensemble individually, BCC samples from the space of possible ensembles, with model weighting randomly drawn from a Dirichlet distribution.

Automatic speech recognition (ASR) can be regarded as a sequential pattern classification problem due to the dynamic nature of the speech signal [92]. The most widely adopted approach to ASR that takes into account both the spectral and temporal information is based on the use of hidden Markov models (HMMs) [14]. Finding the most likely HMM state sequence, referred to as *decoding*, accounts for taking as input the sequence of observed data and yielding as output a corresponding sequence of symbols, such as phonemes or words. As will be discussed in Section 6.2, the well-known plug-in maximum *a posteriori* (MAP) decision rule, e.g., [10, 160], is used to make sequential decisions. In the past, several large vocabulary, continuous speech recognition (LVCSR) [120, 135, 161] systems differing in the feature extraction method, classification

approach, and training algorithm have been developed. They often use complementary information about the speech, but the attempt to preserve one part of the information often leads to the loss of another. As a consequence, each individual ASR approach could only attain a different degree of success. System combination has thus been used to improve the ASR performance, and a lot of techniques is available in the literature. For example, recognizer output voting error reduction (ROVER) combines the outputs of multiple systems into a single network, and voting and/or confidence scores are then used to select the final output [7, 93]. Specifically, ROVER employs the 1-best word sequence from the different systems. These word sequences are aligned, and a decision is made among the words aligned together. This decision can either be based on a simple voting scheme or take confidence scores into account. A limitation inherent in ROVER is the restriction to 1-best word sequences; in fact, only words hypotheses that have been selected by one of the individual systems can be picked in the final results. To augment the space of selectable words hypotheses, word strings can be replaced by confusion networks, and a confusion network combination (CNC) scheme has been proposed in [8]. A confusion network has a fixed number of word positions, and at each position there are a number of alternative words (or the special symbol ? meaning no word is present) [162]. It is possible to show that under reasonable assumptions, the minimum Bayes risk estimate given the confusion network is obtained by taking the most probable symbol (or ϵ, meaning no symbol) at each position [162]. More recently, a refined techniques based on word lattices try to minimize the approximated Bayes risk for system combination was proved to outperform CNC [94], we will refer to this technique as MBR-based combination and use it as term of comparison. The combination can also be accomplished by interpolating scores generated by several systems, e.g., [95–100]. Although those schemes may enhance the ASR accuracy, the fusion technique often does not fully exploit the temporal nature of HMMs. In fact, each individual ASR system generates the sequence of symbols independently, then these sequences are combined in a static fashion.

In [163], a hybrid decoder based on log-likelihood and log-likelihood ratio scores was studied, yet these two scores were obtained from a single base model. To deal with possible mismatches between training and testing speakers [10], a small set of speaker-specific utterances is often collected to update the speaker independent (SI) acoustic model. This process is often referred to as speaker adaptation (SA) [5, 22, 164].

In this chapter, we propose a novel decoding framework by combining K multiple plug-in maximum *a posteriori* (MAP) decoders, where the score combination is carried out at a state level and, over the time. The set of K combination weights is either intuitively chosen or learned from a small amount of adaptation data through a hierarchical Bayesian approach. Deep neural networks (DNNs) [2] and Gaussian mixture models (GMMs) [119] are often used to generate the acoustic phone model scores. Furthermore, SI and SA systems are eventually combined in the proposed plug-in MAP decoder. The experimental evidence on the Switchboard ASR task supports our claims, and up to a statistically significant 12.1% word error rate reduction (WERR) from the top SI baseline systems has been delivered. We would like to remark that the proposed hierarchical Bayesian combination is inspired directly by BCC, yet it exploits the dynamic nature of HMMs more closely to the spirit of structural MAP (SMAP) [22].

6.2 Combining Multiple Plug-in MAP Decoders

A typical decoding problem is to find an unknown sequence of Q symbols, $W = \{w_1, ..., w_Q\}$, from a sequence of T observed feature vectors of a signal, $X = \{o_1, ..., o_T\}$. If the joint distribution, $P(W, X)$, is specified, then it can be considered as solving an optimal decision problem that minimizes the Bayes risk as follows [10, 165]:

$$\hat{W} = arg \max_{W} P(W, X) = arg \max_{W} P(X|W)P(W). \tag{6.2.1}$$

In real-world applications, the two needed distributions, $P(X|W)$ and $P(W)$ are usually unknown and assumed to follow parametric distributions, $P_{\Lambda}(X|W)$ and $P_{\Gamma}(W)$. By plugging the estimated parameters, $\hat{\Lambda}$ and $\hat{\Gamma}$ from training, we end up with the well-known

plug-in MAP decision rule [10],

$$\hat{W} = arg \max_{W} P_{\hat{\Lambda}}(X|W)P_{\hat{\Gamma}}(W). \tag{6.2.2}$$

In some practical situations, the sequence W is usually represented by a set of finer units, called states. So the problem becomes solving a sequence of T states, $S\{s_1, ..., s_T\}$, with the decoded state s_t at stage t associated to observation vector o_t. Now the plug-in MAP decision rule can be expressed as:

$$\hat{S} = arg \max_{S} [\log P_{\hat{\Gamma}}(S) + \sum_{t=1}^{T} \log p_{\hat{\Lambda}}(o_t|s_t)], \tag{6.2.3}$$

in which $P_{\hat{\Gamma}}(S)$ is computed by a state sequence model, often following a Markov chain assumption [166], and $P_{\hat{\Lambda}}(o_t|s_t)$ is evaluated by a state observation model which can usually be characterised with multiple probabilistic assumptions. In this study, we use the same state sequence model for all the K plug-in MAP decoders and focus on combining K multiple state observation models, with the k^{th} model computed by $P_{\hat{\Lambda}_k}(o_t|s_t)$. With w_k as the combination weight for the k^{th} plug-in MAP decoder, we have:

$$\hat{S} = arg \max_{S} [\log P_{\hat{\Gamma}}(S) + \sum_{t=1}^{T} \sum_{k=1}^{K} w_k \log P_{\hat{\Lambda}_k}(o_t|s_t)]. \tag{6.2.4}$$

Note here each decoder has only one single weight w_k. When the states exhibit quite different behaviors from each other, we can use state-specific weight for the state at time t, s_t for the k^{th} decoder, w_{ks_t}, and end up with:

$$\hat{S} = arg \max_{S} [\log P_{\hat{\Gamma}}(S) + \sum_{t=1}^{T} \sum_{k=1}^{K} w_{ks_t} \log P_{\hat{\Lambda}_k}(o_t|s_t)]. \tag{6.2.5}$$

The combination weights, w_{ks_t}, can be intuitively specified in an ad hoc manner, or estimated from a collection of data. We will return to this key issue later.

In ASR, W is usually a sequence of words embedded in an observed spoken utterance, X, o_t is a frame feature vector extracted at time t, s_t is often an HMM state of a phone model, and S is a sequence of phone states, known as senones [29] to characterize the sequence of words, each pronounced as a sequence of phones according to a lexical specification of

the word in a dictionary [167]. In our current setting, we attempt to find the most likely S among all possible word sequences in a search space based on optimizing a combined state log-probabilities in the summation of the LHS of Eqs. (6.2.4) and (6.2.5) above. The decoding process is usually accomplished in a dynamic programming manner [168] with a Viterbi-like decoding algorithm [169]. For each frame o_t, its senone log-likelihoods is often computed by either DNNs or GMMs.

6.3 Hierarchical Bayesian System Combination

In the proposed decoding framework, the combination weights, w_k and w_{ks_t}, mentioned in Section 6.2 can be intuitively specified, or estimated from a collection of data. In this Section, we will elaborate how to learn the combination weights with maximum likelihood (ML) and hierarchical Bayesian estimation.

6.3.1 Combination Weight Estimation

6.3.1.1 Maximum Likelihood (ML) Estimation

The soft weight w_{ks_t} mentioned in Eq. (6.2.5) is what we intend to get next. Here we start from devising a combination scheme through ML estimation of w_k in Eq. (6.2.4).

Let $Z \in \{1, ..., K\}$, be a categorical random variable with K states. Z can only stay in one state indicating which system to choose. With $p(Z = k) = w_k$, we have

$$p(Z|\mathbf{w}) = Mu(Z|\mathbf{w}) = \prod_{k=1}^{K} w_k^{\delta(Z-k)}, \tag{6.3.1}$$

which is called a multinomial distribution, where $0 \leq w_k \leq 1$, $\sum_{k=1}^{K} w_k = 1$ and $\delta(x)$ is a delta function which equals 0 only when $x = 0$.

By observing a sequence of N trials from some training data D, $D = (z_1, ..., z_N)$, of the categorical multinomial random variable Z, corresponding to the state sequence $S = (s_1, ..., s_t, ..., s_N)$ (z_t indicating the choice of candidate systems for state s_t), the likelihood for the sequence is

$$p(z_1, ..., z_N|\mathbf{w}) = \prod_{n=1}^{N} \prod_{k=1}^{K} w_k^{I(x_n=k)} = \prod_{k=1}^{K} w_k^{N_k}, \tag{6.3.2}$$

where $N_k = \sum_{n=1}^{N} I(x_n = k)$. Moreover, The likelihood of a given number of $N_1, ..., N_K$ counts out of N trials is

$$p(N_1, ..., N_K|\mathbf{w}) = \frac{N!}{\prod_{k=1}^{K} N_k!} \prod_{k=1}^{K} w_k^{N_k}, \tag{6.3.3}$$

The maximum likelihood estimation of $\mathbf{w}$ is

$$\hat{\mathbf{w}}^{\mathbf{ML}} = arg \max_{\mathbf{w}} p(z_1, ..., z_N|\mathbf{w}). \tag{6.3.4}$$

We can obtained that

$$\hat{w_k}^{ML} = \frac{N_k}{N}. \tag{6.3.5}$$

To find N_k, for each trial z_t at time t, we look at the target state s_l which satisfies $P_{\Lambda_k}(o_t|s_l) = 1$. $P_{\Lambda_k}(o_t|s_l)$ is obtained from the ground-truth indicating what the correct state should be at time t. Noting that, besides the target state, all other states s with $s \neq s_l$ will have $P_{\Lambda_k}(o_t|s) = 0$. We can obtain the ground-truth choice of candidate systems at time t by

$$z_t = arg \max_{k} P_{\hat{\Lambda}_k}(o_t|s_l). \tag{6.3.6}$$

6.3.1.2 Hierarchical MAP Estimation

Next, state-specific weights are employed as in Eq. (6.2.5), i.e., for each state s, there is a combination weight, w_{ks} in the k^{th} decoder.

In order to do so, we have to accumulate N_{ks} for each individual state s, by assuming there is a corresponding observed sequence of trials for each state s, $D_s(z_{1s}, ..., z_{Ns})$. If we still follows Eq. (6.3.6) accumulating counts only for the target state, for each observation o_t at time t, there will be only one valid trial for one single state, so most of the trials, z_{ts}, will be invalid. As a result, for some state, we might end up with $N_{ks} = 0$ or even $\sum_{k=1}^{K} N_{ks} = 0$.

To deal with such a circumstance, for each observation o_t, besides the target state, we also search for non-target state s ($P_{\Lambda_k}(o_t|s) = 0$) satisfying:

$$\max_{k} P_{\hat{\Lambda}_k}(o_t|s) - \min_{k} P_{\hat{\Lambda}_k}(o_t|s) \geq \theta_s, \tag{6.3.7}$$

where θ_s is a per-state threshold derived from large amount of data (i.e., training data of baseline models) making:

$$P(\max_k P_{\hat{\Lambda}_k}(o_t|s) - \min_k P_{\hat{\Lambda}_k}(o_t|s) \geq \theta_s) = 0.5. \tag{6.3.8}$$

For such states, we treat the z_{ts} as a valid trial, and set

$$z_{ts} = arg\min_k P_{\hat{\Lambda}_k}(o_t|s). \tag{6.3.9}$$

In doing so, we can find valid trials for more states. But $N_{ks} = 0$ or $\sum_{k=1}^{K} N_{ks} = 0$ can still exist. To deal with such circumstance, Bayesian approach can be utilized. For state s, we impose a conjugate Dirichlet prior $Dir(\mathbf{w}_s|\boldsymbol{\alpha}_s)$ for $\mathbf{w}_s$:

$$Dir(\mathbf{w}_s|\boldsymbol{\alpha}_s) = Dir(w_{1s}, ..., w_{Ks}|\alpha_{1s}, ..., \alpha_{Ks}) = \frac{\Gamma(\sum_{k=1}^{K}\alpha_{ks})}{\prod_{k=1}^{K}\Gamma(\alpha_{ks})}\prod_{k=1}^{K} w_{ks}^{\alpha_{ks}-1}, \tag{6.3.10}$$

where the gamma function is defined as

$$\Gamma(x) = \int_0^\infty u^{x-1}e^{-u}du. \tag{6.3.11}$$

If we start with such a Dirichlet prior $Dir(\mathbf{w}_s|\boldsymbol{\alpha}_s)$ and $N_{1s}, ..., N_{Ks}$ are observed, we end up with a Dirichlet posterior $Dir(\mathbf{w}_s|\alpha_{1s} + N_{1s}, ..., \alpha_{Ks} + N_{Ks})$:

$$p(\mathbf{w}|D) = \frac{1}{p(D)}p(D|\mathbf{w})p(\mathbf{w}|\boldsymbol{\alpha}) = Dir(\mathbf{w}|\alpha_1 + N_1, ..., \alpha_K + N_K). \tag{6.3.12}$$

So w_{ks} for choosing the kth system of state s can be estimated as,

$$\hat{w_{ks}}^{MAP} = \frac{\alpha_{ks} + N_{ks}}{\sum_{k=1}^{K}\alpha_{ks} + \sum_{k=1}^{K} N_{ks}}. \tag{6.3.13}$$

In contrast, prior parameters α_{ks}, often known as hyperparameters [170], can be thought of as a pseudo-count and estimated from the estimated posterior density of the parent node of the current state though hierarchical prior evolution [22].

We apply Bayesian learning in a hierarchical way by clustering the states into a tree structure. In the root layer of the tree, all states belong to a single cluster. In the first layer,

the single cluster of the root layer is divided into several smaller clusters. As we go deeper through the tree, finer clusters are obtained by dividing from its parent cluster until the leaf layer is reached in which each leaf represents an individual state.

At the root layer (layer 0), all states are considered belonging to one single group so we accumulate all the valid state trials from all the states for N_k^0. We can obtain $\hat{w}_k^0$ in an ML fashion,

$$\hat{w}_k^0 = \frac{N_k^0}{\sum_{k=1}^K N_k^0}, \tag{6.3.14}$$

where $\hat{w}_k^0$ stands for combination weight for decoder k in the layer 0. Note that the superscripts of the variables indicate the layer in the tree.

In the first layer (layer 1), states are divided into clusters. When we accumulate N_{km}^1 for a cluster m for decoder k, all the valid state trials in cluster m count for that cluster. We then impose the accumulated counts in the root layer, N_k^0, as the prior pseudo counts, i.e., $\alpha_{km}^1 = N_k^0$, then we have $\hat{w}_{km}^1$ by the Bayesian approach,

$$\hat{w}_{km}^1 = \frac{N_k^0 + N_{km}^1}{\sum_{k=1}^K N_k^0 + \sum_{k=1}^K N_{km}^1}. \tag{6.3.15}$$

In the same way, we can accumulate counts and evolve the prior layer by layer until the leaf layer to compute w_{ks}^{leaf} for each individual state.

6.3.2 Combination of Multiple ASR Systems

Applying the proposed decoding framework of combining multiple MAP decoders to ASR, the candidate plug-in MAP decoders for combination are a context-dependent, DNN-HMM (CD-DNN-HMM) system [171], adapted through MAP linear hidden network (LHN) approach [137], and a CD-GMM-HMMs trained over bottleneck features (BN-CD-GMM-HMM) [27, 172] and adapted with SMAP [22]. The two SI plug-in MAP decoders are built with distinct model assumptions and adapted with quite different approaches, so they could be strongly complementary with each other. Nevertheless, the unadapted CD-DNN-HMMs and BN-CD-GMM-HMMs can also serve as candidate systems. Combination weights are learnt speaker specifically using the SA data.

6.3.2.1 CD-DNN-HMMs

DNNs are used to estimate senone emission probabilities [30]. For adaptation of the DNN based ASR system in this work, an augmented affine transform network is inserted into the original neural architecture [77] and MAP adaptation follows [137].

6.3.2.2 BN-CD-GMM-HMMs

Bottleneck features (BN) can be extracted using a small hidden layer, *bottleneck layer*, in the middle of a neural architecture. ASR systems trained over bottleneck features can deliver top recognition performance [158]). Here, SMAP [22] is used to perform speaker adaptation.

6.3.2.3 Decision Tree for Senone Clustering

To perform hierarchical Bayesian estimation of the combination weight as indicated in Section 6.3.1.2, we have to cluster HMM states (i.e., senones) into a tree structure. Senones are obtained through the tree-based state tying approach developed in [173], and that same tree is used here to find the relationship among senones and build our clusters. In this work, the tree is restricted to have three layers: a root, a middle cluster layer, and a leaf layer with all senones.

6.4 Experiments and Result Analysis

6.4.1 Experimental Setups

The baseline SI D0 system in Table 6.4.1 was trained using the 309-hour Switchboard data (SWBD) [135], and the Kaldi toolkit [121], an n-gram language model, trained on SWBD data only [1] was used during decoding. The DNN was initialized with stacked restricted Boltzmann machines (RBMs) [43], by using layer-by-layer generative pre-training. The DNN has six hidden layers, with 2048 units in the first five and 216 units in the last hidden layer (see [137] for details). The output layer has 8806 softmax units corresponding to

[1] Fisher data can also be used to reduce the language model perplexity [112]; however, it is more common to report results without boosting the language model [79, 146].

the senones generated using a CD-GMM-HMM baseline[2]. This DNN configuration is commonly used in the ASR community except for the last hidden BN layer. The input feature vector is a 23-dimension mean-normalized log filter bank feature vector with up to second-order derivatives and a context window of 11 frames, forming a vector of 759-dimension (69×11) input. The DNN was trained with an initial learning rate of 0.008 using the cross-entropy [31, 50, 174, 175] objective function.

The baseline SI G0 system in Table 6.4.1 was trained as described in Section 6.3.2.2 with the BN features generated by the abovementioned baseline. The same set of senones is shared between the D0 and G0 systems. The adaptation and testing data were taken from the NIST 2000 Hub5 evaluation set [156]. We have selected the 40 Switchboard speakers, as in [156], for our experiments, which is roughly 3 minutes of speech material per each speaker. Moreover, we have randomly chosen 20 utterances per speaker to form the adaptation set. The remaining utterances were used in testing. Adaptation experiments were carried out in a supervised manner.

6.4.2 Results and Analysis

6.4.2.1 Fixed-weight Combination of SI Systems

Table 6.4.1 shows the performance of the speaker independent (SI) baseline D0 and G0 systems in the second and third rows, respectively. It is important to emphasise that these two systems attained a recognition accuracy equal to 17.4%, which is comparable to what reported in the recent literature [79,112,146] when (i) the same training material is used, (ii) the same testing conditions are adopted, and (iii) a feed-forward DNN based acoustic model is employed[3]. For the sake of completeness, it should also be pointed out that convolutional neural networks are likely to enhance the recognition accuracy of the baseline systems. However, finding the best configuration for the sizes of filters, pooling, and input feature maps in a convolutional neural network based acoustic model may require an intense and accurate tuning phase, as discussed in [176]. Unfortunately, precise find tuning is out of the

[2]See KALDI documentation for information on senones generation.

[3]The bottleneck layer introduced to generate BN features causes performance differences.

Table 6.4.1. WERs (in %) for speaker independent baseline ASR systems, and fixed-weight system combination. In the parentheses are WERRs.

System	WER
baseline CD-DNN-HMM (D0)	17.4%
baseline BN-CD-GMM-HMM (G0)	17.4%
MBR-based D0 - G0 combination [94]	17.3% (0.6%)
D0 $\oplus$ G0 with fixed w_k = 0.5 (C0)	16.9% (2.9%)
D0 $\oplus$ G0 with learn $w_k s$ with SI training data	**16.8**% (3.4%)

Table 6.4.2. WERs (in %) for speaker adaptive systems with 20 adaptation utterances, and fixed-weight system combination. WERR in parentheses.

System	WER
D0 + MAP-LHN (D1)	16.8% (3.4%)
G0 + SMAP (G1)	16.7% (4.0%)
MBR-based D1 - G1 combination [94]	16.7% (4.0%)
D1 $\oplus$ G1 with fixed w_k = 0.5 (C1)	**16.2**% (6.9%)

scope of the current investigation would deviate us too much from our chief goal, namely proposing an effective system combination approach to handling dynamic patterns, such us senone sequences. Since feed-forward DNN based acoustic models can still be used to attain a competitive recognition performance [112, 177], the assessment of the proposed combining scheme cannot be considered as detrimental to the current investigation and subsequent claims. The next step in our investigation is to assess the performance of

Table 6.4.3. WERs (in %) for hierarchical Bayesian system combination of the systems D0, G0, D1 and G1 from Table 6.4.1 and 6.4.2. In rounded parentheses are WERRs from the D0 and G0 systems in Table 6.4.1.

System	WER
D1 $\oplus$ G1, fixed w_k = 0.5 (C1)	16.2% (6.9%)
D0 $\oplus$ G0 $\oplus$ D1 $\oplus$ G1, fixed w_k (C11)	16.1% (7.5%)
D1 $\oplus$ G1, ML learned w_k at root (C2)	16.0% (8.0%)
D1 $\oplus$ G1, MAP learned w_{ks} at middle (C3)	15.7% (9.8%)
D1 $\oplus$ G1, MAP learned w_{ks} at leaf (C4)	15.5% (10.9%)
D0 $\oplus$ G0 $\oplus$ D1 $\oplus$ G1, MAP learned w_{ks} at leaf (C44)	**15.3**% (12.1%)

several combination schemes. The WER attained with a fixed-weight combination is given in the fifth row in Table 6.4.1. All word error rate reduction (WERR) results reported in rounded parentheses next to the WERs are relative to the SI WER of 17.4% for both the D0

and G0 systems. We can see that the C0 system, a fixed-weight combination of D0 and G0, can provide about a 2.9% WERR from the two baseline systems. A small yet promising further improvement is observed when the combination weights are learnt from training data, and a WER of 16.8% is attained (see sixth row in Table 6.4.1). The latter results may be better appreciated by observing that a more conventional static minimum Bayes risk (MBR)-based system combination approach [94] reported in the fourth row in Table 6.4.1, which takes inspiration from the confusion network combination (CNC) scheme [8] while avoiding its computational complexity, provides a negligible relative improvement of 0.6% over the individual baseline systems.

6.4.2.2 Fixed-weight Combination of SA Systems

Acoustic model adaptation has demonstrated to be a valid and effective approach to modifying the acoustic model parameters to better resemble the evaluation data, as testified by the great deal of DNN adaptation techniques available in the recent literature, e.g., [74, 79, 81, 85, 90, 91, 112–116, 177–179]. The key idea of any adaptation algorithm is like this: starting from a pre-trained (e.g., speaker and/or task independent) speech recognition system, for a new user (or a group of users) to use the system for a specific task, a small amount of *adaptation data* is collected from the user. These data are employed to construct a speaker adaptive system for the speaker in the particular environment for that specific application. By doing so, the mismatch between training and testing can be generally reduced.

In Table 6.4.2, we report results for several speaker-adapted (SA) systems. We can observe a WER reduction by applying MAP LHN adaptation [177] to D0 (D1 in the second row). The third row shows that the SMAP-adapted G1 systems also attains a better performance of the speaker independent G0 seed system. The combined speaker adapted system, denoted as C1, is shown in the bottom row. We can see that the D1 and G1 systems both achieved about ~3.5-4% WERR. Again, the fixed-weight combination approach is shown to be effective by providing a WERR of ~7%. On the other hand, a

static MBR-based system combination attains a final performance that equals that of the SMAP-adapted G1 system, as shown in the fourth row in Table 6.4.2.

6.4.2.3 Hierarchical Bayesian Combination with Speaker-dependent, Senone-specific Weights

The WERs attained with the proposed hierarchical Bayesian weights are given in Table 6.4.3. The second row in Table 6.4.3 shows again the WER attained with the C1 combination scheme for ease of comparison. The third row shows the result by the combination of 4 systems, namely D0, G0, D1, G1 with fixed weight $w = 0.05, 0.05, 0.45, 0.45$, respectively. From the fourth to the sixth row, the performances of combining two systems, D1 and G1, attained in root, middle and leaf layer are shown, respectively. The last row shows the WER attained by combining 4 systems in leaf layer. The best results given by hierarchical Bayesian combination achieve a WERR of 10.9% and 12.1% from the top SI performance of 17.4% obtained by D0 and G0 by combining 2 and 4 systems in C4 and C44 shown in Table 6.4.3, respectively. Moreover a WERR of 7.2% and 8.4% from the top SA WER of 16.7% obtained by G1 as shown in Table 6.4.2 are achieved by combining 2 and 4 systems, respectively. The experimental evidence reported in this section clearly demonstrate not only the the effectiveness of the proposed hierarchical Bayesian combination approach, but that dynamic system combination is indeed the best way to deploy combination of plug-in maximum a posteriori decoders.

6.5 Summary

In this chapter, we have proposed a hierarchical Bayesian system combination framework by combining the frame-level acoustic probability scores evaluated at the HMM state level of DNN and GMM models. We demonstrate the effectiveness of the proposed approach through experimental results obtained from the Switchboard LVCSR task. We choose the CD-DNN-HMM and BN-CD-GMM-HMM systems as combination candidates because they are complementary to each other in evaluating the frame likelihoods.The proposed

scheme perform system combination in a dynamic way exploiting the temporal nature of HMMs thus supports real-time multi-stream speech recognition. In the future, we would like to further extend the flexibility of the combination framework by introducing other conjugate priors, such as mixture of Dirichlets or even non-conjugate ones, and try the system combination strategy in other areas, such as computer vision and natural language processing applications.

CHAPTER 7

CONCLUSION

In this dissertation, we try to deploy a Bayesian adaptation/combination framework for deep model based ASR systems to combat the degradation of the recognition accuracy, which is typically observed under potential mismatched conditions between training and testing. We followed the three directions laid out in the Introduction chapter and made the following contributions in this dissertation work.

For the first direction about performing Bayesian adaptation directly on the discriminative DNN models, maximum a posteriori estimation is employed in the manner of regularization in the DNN updating formula. For the supervised batch adaptation, we build the adaptation technology based on MAP estimation of the augmented linear hidden network (LHN) parameters. The proposed MAP adaptation scheme can provide a substantial relative word error rate (WER) reduction from an already-strong speaker independent CD-DNN-HMM baseline and consistently outperform conventional transformation based adaptation schemes. Then a hierarchical adaptation technique is proposed through multi-task learning (MTL) [24] devising regularization provided by auxiliary tasks. By adding auxiliary architecture to the original DNN and performing MTL with the secondary tasks, we improve the learning ability of the original DNN structure and thus enhance the discrimination power of the DNN models with limited adaptation data. To make the technique applicable to practical situations, we apply it on unsupervised online adaptation. To deal with the data scarcity problem, which becomes more severe in online adaptation, we further reduce the parameter number by applying the MAP adaptation framework on the activation function parameters only, instead of adapting the augmented LHN parameters. The proposed Bayesian framework's adaptive nature for performing iterative learning provides promising online adaptation results. Direct adaptation on DNN has the advantage of easy deployment, so there are a lot of

on-going efforts in this direction directly following what is proposed in Section 3.2 (i.e., [180]) and Section 3.3 (i.e., [181]) and beyond [182–184]. Yet this dissertation represents the first Bayesian effort on direct deep model adaptation. Proposed framework can be applied to different kinds of deep models and different applications.

For the second direction on casting DNN into a generative framework to better leverage Bayesian based technologies for adaptation, we explore different ways for DNNs' generative casting and try to utilize classic Bayesian techniques to perform adaptation on the generative models derived from the DNNs. In our first attempt, we focus on the adaptation of weights and biases in the last DNN affine transformation layer and devise a maximum likelihood (ML) estimation solution to perform model adaptation. Next, a MAP adaptation scheme is formulated by incorporating prior information obtained by applying the proposed ML approach with the training data. Experimental results show that the proposed approach gives better results than the conventional transformation based approaches applied to the same parameters. Unfortunately, MAP is not always able to further improve the adaptation results. In the second attempt, we go through an indirect way by converting the discriminative CD-DNN-HMMs to the generative CD-GMM-HMMs. We are able to build deep BN-CD-GMM-HMMs (BN features based CD-GMM-HMMs) with comparable performance with CD-DNN-HMMs by using the BN features [27] derived from the CD-DNN-HMMs. Then the Bayesian adaptation is done by applying the original MAP/SMAP technologies to the BN-CD-GMM-HMMs. Experimental results show the effectiveness of the proposed indirect Bayesian method and also demonstrate the ability of DNN to serve as a bridge function between the data distribution and the GMM based decision model [28]. The BN-CD-GMM-HMMs have a lot of advantage over the CD-DNN-HMMs. One of the most important strengths is that with the GMM based BN-CD-GMM-HMMs we can take advantage of the classic statistical machine learning techniques including the Bayesian family more easily. Now the importance of BN-CD-GMM-HMMs is being gradually

recognized by the community. Recent efforts on improving the learning paradigm of BN-CD-GMM-HMMs are represented in [185].

In the third direction, by leveraging the complementarity of the discriminative and generative adaptive models, a hierarchical Bayesian system combination technique is employed to further enhance the adaptation performance. The CD-DNN-HMMs adapted with MAP-LHN in the direct manner and the BN-CD-GMM-HMMs adapted with SMAP in the indirect manner utilize different model assumptions and are adapted with highly different techniques, and as a result, their potential to have complementarity is promising. A novel system combination scheme is proposed to leverage the complementarity for the improvement of the adaptation performance. In the proposed system combination scheme, per-speaker, per-senone combination weights are learned from the adaptation data through the proposed hierarchical Bayesian approach. Experimental results show the effectiveness of the proposed method by providing nice WERRs on the already adapted baseline deep models. The proposed scheme performs the system combination in a dynamic way so that it can exploit the temporal nature of HMMs thus supports real-time multi-stream speech recognition.

The Bayesian adaptation proposed in this dissertation could be the first step in the integration of the Bayesian techniques and deep neural networks. For future perspective, since the proposed Bayesian adaptation framework mainly relies on Bayesian point estimation, extending it to variational Bayesian inference can be an interesting direction. For practical use, applying the proposed Bayesian adaptation framework to different kinds of deep models, such as long-short-term-memory based recurrent neural networks and convolutional neural networks can also be of great potential.

REFERENCES

[1] K. H. Davis, R. Biddulph, and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 637–642, 1952.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, A. Sehr, W. Kellermann, and R. Maas, "The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, 2013, pp. 1–4.

[4] P. Price, W. M. Fisher, J. Bernstein, and D. S. D. S. Pallett, "The darpa 1000-word resource management database for continuous speech recognition," in *Proc. ICASSP*, 1988, pp. 651–654.

[5] J. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Trans. Speech and audio processing*, vol. 2, no. 2, pp. 291–298, 1994.

[6] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer, Speech, and Language*, vol. 12, pp. 75–98, 1998.

[7] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *Proc. ASRU*, 1997, pp. 347–354.

[8] G. Evermann and P. C. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *Proc. Speech Transcription Workshop*, vol. 27, 2000.

[9] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis*. Wiley New York, 1973, vol. 2.

[10] C.-H. Lee and Q. Huo, "On adaptive decision rules and decision parameter adaptation for automatic speech recognition," *Proc. IEEE*, vol. 88, no. 8, 2000.

[11] L. R. Rabiner and B.-H. Juang, *Fundamentals of speech recognition*. PTR Prentice Hall, 1993.

[12] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE trans. on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[13] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for lvcsr." in *Proc. INTERSPEECH*, 2014, pp. 890–894.

[14] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[15] B.-H. Juang and L. R. Rabiner, "Hidden markov models for speech recognition," *Technometrics*, vol. 33, no. 3, pp. 251–272, 1991.

[16] F. Jelinek, "Interpolated estimation of markov source parameters from sparse data," in *Proc. Workshop on Pattern Recognition in Practice*, 1980.

[17] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE trans. acoustics, speech, and signal processing*, vol. 35, no. 3, pp. 400–401, 1987.

[18] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text compression*. Prentice-Hall, Inc., 1990.

[19] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. ICASSP*, vol. 1, 1995, pp. 181–184.

[20] G. Saon, H.-K. J. Kuo, S. Rennie, and M. Picheny, "The ibm 2015 english conversational telephone speech recognition system," in *Proc. INTERSPEECH*, 2016.

[21] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. INTERSPEECH*, vol. 2, 2010, p. 3.

[22] K. Shinoda and C.-H. Lee, "A structural Bayes approach to speaker adaptation," *IEEE trans. Speech and Audio Processing*, vol. 9, no. 3, pp. 276–287, 2001.

[23] S. Watanabe, Y. Minami, A. Nakamura, and N. Ueda, "Application of variational Bayesian estimation and clustering to acoustic model adaptation," in *Proc. ICASSP*, vol. 1, 2003, pp. I–568.

[24] R. Caruna, "Multitask learning: A knowledge-based source of inductive bias," in *Proc. ICML*, 1993, pp. 41–48.

[25] A. M. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE journal*, vol. 37, no. 2, pp. 233–243, 1991.

[26] H. Hermansky, D. P. Ellis, and S. Sharma, "Tandem connectionist feature extraction for conventional hmm systems," in *Proc. ICASSP*, vol. 3, 2000, pp. 1635–1638.

[27] F. Grézl, M. Karafiát, S. Kontár, and J. Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc. ICASSP*, vol. 4, 2007, pp. IV–757.

[28] B.-H. Juang, "Deep neural networks–a developmental perspective," *APSIPA Transactions on Signal and Information Processing*, vol. 5, p. e7, 2016.

[29] M.-Y. Hwang and X. Huang, "Shared-distribution hidden Markov models for speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 1, no. 4, pp. 414–420, 1993.

[30] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[31] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Found. Trends Signal Process.*, vol. 7, no. 3-4, pp. 197–387, 2014.

[32] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[33] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[34] A. G. I. V. G. Lapa, "Cybernetic predicting devices," CCM Information Corporation, Tech. Rep., 1965.

[35] A. G. Ivakhnenko, "The group method of data handling-a rival of the method of stochastic approximation," *Soviet Automatic Control*, vol. 13, no. 3, pp. 43–55, 1968.

[36] R. Dechter, *Learning while searching in constraint-satisfaction problems*. University of California, Computer Science Department, Cognitive Systems Laboratory, 1986.

[37] I. N. Aizenberg, N. N. Aizenberg, and A. G. Krivosheev, "Multi-valued and universal binary neurons: learning algorithms, application to image processing and recognition," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. Springer, 1999, pp. 21–35.

[38] A. E. Bryson, "A gradient method for optimizing multi-stage allocation processes," in *Proc. Harvard Univ. Symposium on digital computers and their applications*, 1961, p. 72.

[39] H. J. Kelley and H. J, "Gradient theory of optimal flight paths," *Ars Journal*, vol. 30, no. 10, pp. 947–954, 1960.

[40] S. Dreyfus, "The numerical solution of variational problems," *Journal of Mathematical Analysis and Applications*, vol. 5, no. 1, pp. 30–45, 1962.

[41] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning representations by back-propagating errors*. MIT Press, Cambridge, MA, USA, 1988.

[42] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[43] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. CVPR*, 2015.

[46] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[47] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Springer, 1994, vol. 247.

[48] J. Pan, C. Liu, Z. Wang, Y. Hu, and H. Jiang, "Investigation of deep neural networks (dnn) for large vocabulary continuous speech recognition: Why dnn surpasses gmms in acoustic modeling," in *Proc. International Symposium on Chinese Spoken Language Processing*, 2012, pp. 301–305.

[49] B. Kingsbury, T. N. Sainath, and H. Soltau, "Scalable minimum Bayes risk training of deep neural network acoustic models using distributed Hessian-free optimization." in *Proc. INTERSPEECH*, 2012.

[50] S. Wiesler, J. Li, and J. Xue, "Investigations on Hessian-free optimization for cross-entropy training of deep neural networks." in *Proc. INTERSPEECH*, 2013, pp. 3317–3321.

[51] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, 2013.

[52] M. Cai, Y. Shi, and J. Liu, "Deep maxout neural networks for speech recognition," in *Proc. ASRU*, 2013, pp. 291–296.

[53] P. Swietojanski, J. Li, and J.-T. Huang, "Investigation of maxout networks for speech recognition," in *Proc. ICASSP*, 2014.

[54] Z. Huang, J. Li, C. Weng, and C.-H. Lee, "Beyond cross-entropy: Towards better frame-level objective functions for deep neural network training in automatic speech recognition," in *Proc. INTERSPEECH*, 2014.

[55] K. Veselỳ, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. INTERSPEECH*, 2013, pp. 2345–2349.

[56] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *Proc. ICASSP*, 2012, pp. 4277–4280.

[57] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. ICASSP*, 2013, pp. 8614–8618.

[58] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *Proc. INTERSPEECH*, 2014, pp. 338–342.

[59] D. Yu and M. L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proc. INTERSPEECH*, 2011, pp. 237–240.

[60] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[61] L. E. Baum, "An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, vol. 3, pp. 1–8, 1972.

[62] L. R. Rabiner, J. G. Wilpon, and B.-H. Juang, "A segmental k-means training procedure for connected word recognition," *AT&T technical journal*, vol. 65, no. 3, pp. 21–31, 1986.

[63] Q. Huo and C.-H. Lee, "On-line adaptive learning of the continuous density hidden Markov model based on approximate recursive Bayes estimate," *IEEE trans. Speech and Audio Processing*, vol. 5, no. 2, pp. 161–172, 1997.

[64] ——, "On-line adaptive learning of the correlated continuous density hidden Markov models for speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 6, no. 4, pp. 386–397, 1998.

[65] M. J. F. Gales and P. C. Woodland, "Mean and variance adaptation within the MLLR framework," *Computer Speech & Language*, vol. 10, no. 4, pp. 249–264, 1996.

[66] T. Anastasakos and S. V. Balakrishnan, "The use of confidence measures in unsupervised adaptation of speech recognizers." in *Proc. ICSLP*, 1998.

[67] M. Padmanabhan, G. Saon, and G, "Lattice-based unsupervised mllr for speaker adaptation," in *Proc. ISCA ITRW ASR2000*, 2000.

[68] L. F. Uebel and P. C. Woodland, "Speaker adaptation using lattice-based MLLR," in *Proc. ISCAITR-Workshop on Adaptation Method for Speech Recognition*, 2001.

[69] C. Chesta, O. Siohan, and C.-H. Lee, "Maximum a posteriori linear regression for hidden Markov model adaptation," in *Proc. Eurospeech*, 1999.

[70] A. Acero, L. Deng, T. T. Kristjansson, and J. Zhang, "Hmm adaptation using vector taylor series for noisy speech recognition." in *Proc. INTERSPEECH*, 2000, pp. 869–872.

[71] Y. Zhao and B.-H. Juang, "A comparative study of noise estimation algorithms for VTS-based robust speech recognition." in *Proc. INTERSPEECH*, 2010, pp. 2090–2093.

[72] K. Kalgaonkar and M. A. Clements, "HMM adaptation using sparse probabilistic space mapping for noisy speech," in *Proc. ICASSP*, 2010, pp. 4554–4557.

[73] M. Franch, "Catastrophic forgetting in connectionist networks: causes, consequences and solutions," *Trends in Cognitive Sciences*, vol. 3, no. 4, 1994.

[74] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *Proc. ICASSP*, 2013, pp. 7893–7897.

[75] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *Proc. ICASSP*, 2013, pp. 7947–7951.

[76] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system," in *Proc. Eurospeech*, 1995.

[77] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. D. Mori, "Linear hidden transformations for adaptation of hybrid ANN/HMM models," *Speech Communication*, vol. 49, no. 10, pp. 827–835, 2007.

[78] B. Li and K. C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems," in *Proc. INTERSPEECH*, 2010, pp. 526–529.

[79] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, 2011, pp. 24–29.

[80] K. Yao, D. Yu, F. Seide, H. Su, L. Deng, and Y. Gong, "Adaptation of context-dependent deep neural networks for automatic speech recognition," in *Proc. Spoken Language Technology Workshop*, 2012, pp. 366–369.

[81] J. Li, J.-T. Huang, and Y. Gong, "Factorized adaptation for deep neural network," in *Proc. ICASSP*, 2014.

[82] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[83] P. J. Moreno, "Speech recognition in noisy environments," Ph.D. dissertation, Carnegie Mellon University Pittsburgh, 1996.

[84] J. Li, L. Deng, D. Yu, Y. Gong, and A. Acero, "A unified framework of hmm adaptation with joint compensation of additive and convolutive distortions," *Computer Speech & Language*, vol. 23, no. 3, pp. 389–405, 2009.

[85] S. M. Siniscalchi, J. Li, and C.-H. Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2152–2161, 2013.

[86] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. STL*, 2014.

[87] C. Zhang and P. C. Woodland, "DNN speaker adaptation using parameterised sigmoid and ReLU hidden activation functions," in *Proc. ICASSP*, 2016, pp. 5300–5304.

[88] J. Xue, J. Li, D. Yu, M. Seltzer, and Y. Gong, "Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network," in *Proc. ICASSP*, 2014.

[89] S. Xue, H. Jiang, and L. Dai, "Speaker adaptation of hybrid NN/HMM model for speech recognition based on singular value decomposition," in *Proc. International Symposium on Chinese Spoken Language Processing*, 2014.

[90] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. ASRU*, 2013, pp. 55–59.

[91] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *Proc. ICASSP*, 2013, pp. 7942–7946.

[92] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE Trans. Audio, Speech and Lang. Process.*, vol. 21, pp. 1060–1089, 2013.

[93] H. Schwenk and J.-L. Gauvain, "Combining multiple speech recognizers using voting and language model information." in *Proc. INTERSPEECH*, 2000, pp. 915–918.

[94] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum Byes risk decoding and system combination based on a recursion for edit distance," *Computer Speech & Language*, vol. 25, no. 4, pp. 802–828, 2011.

[95] K. Kirchhoff, G. A. Fink, and G. Sagerer, "Conversational speech recognition using acoustic and articulatory input," in *Proc. ICASSP*, vol. 3, 2000, pp. 1435–1438.

[96] A. Morris, A. Hagen, H. Glotin, and H. Bourlard, "Multi-stream adaptive evidence combination for noise robust ASR," *Speech Communication*, vol. 34, no. 1, pp. 25–40, 2001.

[97] F. Metze and A. Waibel, "A flexible stream architecture for ASR using articulatory features," in *Proc. INTERSPEECH*, Denver, USA, Sept. 2002, pp. 16–20.

[98] S. M. Siniscalchi and C.-H. Lee, "A study on integrating acoustic-phonetic information into lattice rescoring for automatic speech recognition," *Speech Communication*, vol. 51, pp. 1139–1153, 2009.

[99] P. Swietojanski, A. Ghoshal, and S. Renals, "Revisiting hybrid and gmm-hmm system combination techniques," in *Proc. ICASSP*, 2013, pp. 6744–6748.

[100] L. Deng and J. C. Platt, "Ensemble deep learning for speech recognition." in *Proc. INTERSPEECH*, 2014, pp. 1915–1919.

[101] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[102] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, March 1998.

[103] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[104] ——, "Random forests," *Machine Learning*, vol. 45, p. 2001, 2001.

[105] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. of International Conference on Machine Learning*, 1996, pp. 148–156.

[106] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259, 1992.

[107] J. M. Bernardo and A. F. M. Smith, *Bayesian theory*. Wiley, 1994.

[108] Z. Ghahramani and H.-C. Kim, "Bayesian model combination," Gatsby Computational Neuroscience Unit, University College London, Tech. Rep., 2003.

[109] C.-H. Lee, "On stochastic feature and model compensation approaches to robust speech recognition," *Speech Communication*, vol. 25, no. 1-3, pp. 29–47, 1998.

[110] S. M. Siniscalchi, D.-C. Lyu, T. Svendsen, and C.-H. Lee, "Experiments on cross-language attribute detection and phone recognition with minimal target-specific training data," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 20, pp. 875–887, 2012.

[111] R. Price, I. Kenichi, and K. Shinoda, "Speaker adaptation of deep neural networks using a hierarchy of output layers," in *Proc. SLT*, 2014.

[112] P. Swietojanski, J. Li, and S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 24, pp. 1450–1463, 2016.

[113] C. Wu and M. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition," in *Proc. ICASSP*, 2015, pp. 4315–4319.

[114] U. Remes, A. R. López, and D. Palom´aki, "Bounded conditional mean imputation with observation uncertainties and acoustic model adaptation," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 23, pp. 1198–1208, 2015.

[115] L. Samarakoon and K. C. Sim, "Factorized hidden layer adaptation for deep neural network based acoustic modeling," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 24, pp. 2241–2250, 2016.

[116] P. Swietojanski and S. Renals, "Differentiable pooling for unsupervised acoustic model adaptation," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 24, pp. 1773–1784, 2016.

[117] O. Abdel-Hamid and H. Jiang, "Rapid and effective speaker adaptation of convolutional neural network based models for speech recognition," in *Proc. INTERSPEECH*, 2013, pp. 1248–1252.

[118] H. Robbins, "The empirical Bayes approach to statistical decision problems," *Ann. Math. Stat.*, vol. 35, no. 1, 1964.

[119] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[120] D. B. Paul and J. M. Baker, "The design for the wall street journal-based CSR corpus," in *Proc. Workshop on Speech and Natural Language*, 1992, pp. 899–902.

[121] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovskỳ, G. Stemmer, and K. Veselỳ, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[122] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *Proc. ICASSP*, 2013, pp. 6655–6659.

[123] J. Xue, J. Li, and Y. Gong, "Restructuring of deep neural network acoustic models with singular value decomposition." in *INTERSPEECH*, 2013, pp. 2365–2369.

[124] Z. Huang, J. Li, S. M. Siniscalchi, I.-F. Chen, C. Weng, and C.-H. Lee, "Feature space maximum a posteriori linear regression for adaptation of deep neural networks," in *Proc. INTERSPEECH*, 2014, pp. 2992–2996.

[125] N. Srivastava and R. Salakhutdinov, "Discriminative transfer learning with tree-based priors," in *Proc. NIPS*, 2013.

[126] X. Li and X. Wu, "Decision tree based state tying for speech recognition using DNN derived embeddings," in *Proc. International Symposium on Chinese Spoken Language Processing*, 2014, pp. 123–127.

[127] S. Parveen and P. Green, "Multitask learning in connectionist robust asr using recurrent neural networks." in *Proc. INTERSPEECH*, 2003.

[128] Y. Lu, F. Lu, S. Sehgal, S. Gupta, J. Du, C. Tham, P. Green, and V. Wan, "Multitask learning in connectionist speech recognition," in *Proc. Australian International Conference on Speech Science and Technology*, 2004.

[129] M. Seltzer and J. Droppo, "Multi-task learning in deep neural networks for improved phoneme recognition," in *Proc. ICASSP*, 2013, pp. 6965–6969.

[130] Z. Wen, K. Li, Z. Huang, J. Tao, and C.-H. Lee, "Learning auxiliary categorical information for speech synthesis based on deep and recurrent neural networks," in *Proc. International Symposium on Chinese Spoken Language Processing*, 2016.

[131] Y. Xu, J. Du, Z. Huang, L.-R. Dai, and C.-H. Lee, "Multi-objective learning and mask-based post-processing for deep neural network based speech enhancement," in *Proc. INTERSPEECH*, 2015.

[132] G. Heigold, H. Ney, P. Lehnen, T. Gass, and R. Schluter, "Equivalence of generative and log-linear models," *IEEE Trans. Audio, Speech & Language Processing*, vol. 19, no. 5, pp. 1138–1148, 2011.

[133] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size DNN with output-distribution-based criteria," in *Proc. INTERSPEECH*, 2014.

[134] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proc. Assoc. Comput. Linguist.*, 1995, pp. 198–196.

[135] J. J. Godfrey and E. Holliman, "Switchboard-1 release 2," *Linguistic Data Consortium, Philadelphia*, 1997.

[136] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. ICASSP*, 2009, pp. 3761–3764.

[137] Z. Huang, S. M. Siniscalchi, I.-F. Chen, , J. Li, J. Wu, and C.-H. Lee, "Maximum a posteriori adaptation of network parameters in deep models," in *INTERSPEECH*, 2015, pp. 1076–1080.

[138] A. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Feedforward sequential memory networks: A new structure to learn long-term dependency," in *arXiv preprint arXiv:1512.08301*, 2015.

[139] S. Hochreiter and J. Schmidhuber, "Long short- term memory. neural computation," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[140] A. Graves, A. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[141] H. Sak, A. Senior, and B. F, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. INTERSPEECH*, 2014, pp. 338–342.

[142] T. N. Sainath, A. M. B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. ICASSP*, 2013, pp. 8614–8618.

[143] V. V. Digalakis, D. Rtischev, and L. G. Neumeye, "Speaker adaptation using constrained estimation of gaussian mixtures," *IEEE Trans. Speech Audio Process.*, vol. 3, no. 4, pp. 357–366, Sept. 1995.

[144] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech and Language Processing*, vol. 19, pp. 788–798, 2011.

[145] Y. Miao, H. Zhang, and F. Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 23, pp. 1938–1949, 2015.

[146] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE/ACM Trans. on Audio, Speech and Lang. Proc.*, vol. 22, no. 12, pp. 1713–1725, 2014.

[147] A. Mohamed, T. Sainath, G. D. B. Ramabhadran, G. E. Hinton, and M. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proc. ICASSP*, 2011, pp. 5060–5063.

[148] Y. Zhao, J. Li, J. Xue, and Y. Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data," in *Proc. ICASSP*, 2015, pp. 4310–4314.

[149] C. Zhang and P. C. Woodland, "DNN speaker adaptation using parameterised sigmoid and relu hidden activation functions," in *Proc. ICASSP*, 2016, pp. 5300–5304.

[150] M. Karafiát, L. Burget, P. Matějka, O. Glembek, and J. Cernocky, "iVector-based discriminative adaptation for automatic speech recognition," in *Proc. ASRU*, 2011, pp. 152–157.

[151] P. Karanasou, Y. Wang, M. Gales, and P. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proc. INTERSPEECH*, 2014, pp. 2180–2184.

[152] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Macmillan, 1994.

[153] J. Kaiser, B. Horvat, and Z. Kačič, "A novel loss function for the overall risk criterion based discriminative training of HMM models," in *Proc. ICSLP.*, 2000, pp. 887–890.

[154] M. Gibson and T. Hain, "Hypothesis spaces for minimum Bayes risk training in large vocabulary speech recognition," in *Proc. INTERSPEECH*, 2006, pp. 2406–2409.

[155] M. H. DeGroot, *Optimal Statistical Decisions*. McGraw-Hill, New York, USA, 1970.

[156] J. Fiscus, W. M. Fisher, A. F. Martin, M. A. Przybocki, and D. S. Pallett, "2000 NIST evaluation of conversational speech recognition over the telephone: English and Mandarin performance results," in *Proc. Speech Transcription Workshop*, 2000.

[157] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks." in *Proc. INTERSPEECH*, 2011, pp. 437–440.

[158] Z. Yan, Q. Huo, and J. Xu, "A scalable approach to using DNN-derived features in GMM-HMM based acoustic modeling for LVCSR." in *Proc. INTERSPEECH*, 2013, pp. 104–108.

[159] M. J. F. Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Trans. Speech and Audio Processing*, vol. 7, no. 3, 1999.

[160] C.-H. Lee and S. M. Siniscalchi, "An information-extraction approach to speech processing: Analysis, detection, verification, and recognition," *Proc. IEEE*, vol. 101, no. 5, pp. 1089–1115, May 2013.

[161] J.-L. Gauvain and L. Lamel, "Large vocabulary continuous speech recognition: Advances and applications," *Proc. IEEE*, vol. 88, no. 8, pp. 1181–1200, 2000.

[162] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[163] M.-W. Koo, C.-H. Lee, and B.-H. Juang, "Speech recognition and utterance verification based on a generalized confidence score," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 8, pp. 821–831, 2001.

[164] C.-H. Lee, C.-H. Lin, and B.-H. Juang, "A study on speaker adaptation of the parameters of continuous density hidden Markov models," *IEEE Trans. Signal Processing*, vol. 39, no. 4, pp. 806–814, 1991.

[165] A. Nadas, "Optimal solution of a training problem in speech recognition," *IEEE Trans. Acoustics, Speech and Signal processing*, vol. 33, no. 1, pp. 326–329, 1985.

[166] J. R. Norris, *Markov chains*. Cambridge university press, 1998, no. 2.

[167] L. Lamel and G. Adda, "On designing pronunciation lexicons for large vocabulary continuous speech recognition," in *Proc. ICSLP*, vol. 1, 1996, pp. 6–9.

[168] H. Ney and S. Ortmanns, "Progresses in dynamic programming search for LVCSR," *Proc. IEEE*, vol. 88, no. 8, pp. 1224–1240, 2000.

[169] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[170] M. H. DeGroot and H. Morris, *Optimal statistical decisions*. John Wiley & Sons, 2005, vol. 82.

[171] H. Bourlard and N. Morgan, *Connectionist speech recognition: A hybrid approach*. Kluwer Academic Publishers, 1994.

[172] D. Yu and M. Seltzer, "Improved bottleneck features using pretrained deep neural networks." in *Proc. INTERSPEECH*, vol. 237, 2011, p. 240.

[173] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, 1994, pp. 307–312.

[174] P. T. D. Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.

[175] R. A. Dunne and N. A. Campbell, "On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function," in *Proc. 8th Aust. Conf. on the Neural Networks*, vol. 185, 1997.

[176] Y. Qian, M. Bi, T. Tan, and K. Yu, "Very deep convolutional neural networks for noise robust speech recognition," *IEEE/ACM Trans. Audio Speech and Language Processing*, vol. 24, no. 12, pp. 2263–2276, 2016.

[177] Z. Huang, S. M. Siniscalchi, and C.-H. Lee, "Bayesian unsupervised batch and online speaker adaptation of activation function parameters in deep models for automatic speech recognition," *IEEE/ACM Trans. Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 64–75, 2017.

[178] R. Price, K.-I. Iso, and K. Shinoda, "Speaker adaptation of deep neural networks using a hierarchy of output layers," in *Proc. STL*, 2014.

[179] Z. Huang, J. Li, S. M. Siniscalchi, I.-F. Chen, J. Wu, and C.-H. Lee, "Rapid adaptation for deep neural networks through multi-task learning," in *Proc. INTERSPEECH*, 2015.

[180] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabisa, C. Clopathb, D. Kumarana, and R. Hadsella, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, 2017.

[181] C.-K. Wei, C.-T. Chung, H.-Y. Lee, and L.-S. Lee, "Personalized acoustic modeling by weakly supervised multi-task deep learning using acoustic tokens discovered from unlabeled data," in *Proc. ICASSP*, 2017.

[182] L. Samarakoon, K. C. Sim, and B. Mak, "An investigation into learning effective speaker subspaces for robust unsupervised dnn adaptation," in *Proc. ICASSP*, 2017.

[183] Y. Zhao, J. Li, K. Kumar, and Y. Gong, "Extended low-rank plus diagonal adaptation for deep and recurrent neural networks," in *Proc. ICASSP*, 2017.

[184] Z.-Q. Wang and D. Wang, "Unsupervised speaker adaptation of batch normalized acoustic models for robust as," in *Proc. ICASSP*, 2017.

[185] C. Zhang and P. C. Woodland, “Joint optimisation of tandem systems using gaussian mixture density neural network discriminative sequence training,” in *Proc. ICASSP*, 2017.

VITA

Zhen Huang is currently pursuing a Ph.D. degree under the supervision of Prof. Chin-Hui Lee at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. He received his bachelor's degree in Electrical&Computer Engineering from Southeast University, Nanjing, China, in 2009 and received dual master's degrees in Electrical&Computer Engineering from Shanghai JiaoTong University, Shanghai, China and Georgia Institute of Technology, Atlanta, GA, in 2012. The publications during his Ph.D. study in Georgia Institute of Technology are listed in the following.

(1) Z. Huang, S. M. Siniscalchi, C.-H. Lee, "Bayesian Unsupervised Batch and Online Speaker Adaptation," IEEE/ACM Trans. on Audio, Speech, and Language Processing 2017

(2) B. Wu, K. Li, F. Ge, Z. Huang, M. Yang, S. M. Siniscalchi and C.-H. Lee, "An End-to-End Deep Learning Approach to Simultaneous Speech Dereverberation and Acoustic Modeling for Robust Speech Recognition," submitted to IEEE Journal of Selected Topics in Signal Processing, 2017

(3) Z. Huang, S. M. Siniscalchi, C.-H. Lee, "Hierarchical Bayesian Combination of Plug-in Maximum A Posteriori Decoders in DNN-based Speech Recognition and Speaker Adaptation," submitted to Pattern Recognition Letters, 2017

(4) Z. Huang, S. M. Siniscalchi, C.-H. Lee, "A Unified Approach to Transfer Learning of Deep Neural Networks with Applications to Speaker Adaptation in Automatic Speech Recognition," Neurocomputing 2016

(5) Z. Wen, K. Li, Z. Huang, J. Tao and C.-H. Lee, "Learning Auxiliary Categorical Information For Speech Synthesis Based On Deep And Recurrent Neural Networks," International Symposium on Chinese Spoken Language Processing, 2016

(6) B. Wu, K. Li, Z. Huang, S. M. Siniscalchi, M. Yang and C.-H. Lee, "A Unified Deep Modeling Approach to Simultaneous Speech Dereverberation and Recognition for the Reverb Challenge," Joint Workshop on Hands-free Speech Communication and Microphone Arrays, 2017

(7) S. Wang, K. Li, Z. Huang, S. M. Siniscalchi and C.-H. Lee, "A Transfer Learning and Progressive Stacking Approach to Reducing Deep Model Sizes with an Application to Speech Enhancement," ICASSP 2017

(8) Z. Huang, S. M. Siniscalchi, C.-H. Lee, "Towards A Direct Bayesian Adaptation Framework for Deep Models," Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, 2016

(9) K. Li, Z. Huang, Y. Xu and C.-H. Lee, "DNN-Based Speech Bandwidth Expansion and Its Application to Adding High-Frequency Missing Features for Automatic Speech Recognition of Narrowband Speech," INTERSPEECH 2015

(10) Y. Xu, J. Du, Z. Huang, L.-R. Dai and C.-H. Lee, "Multi-Objective Learning and Mask-Based Post-Processing for Deep Neural Network Based Speech Enhancement," INTERSPEECH 2015

(11) Z. Huang, S. M. Siniscalchi, I-F. Chen, J. Li, J. Wu and C.-H. Lee, "Maximum a Posteriori Adaptation of Network Parameters in Deep Models," INTERSPEECH 2015

(12) Z. Huang, J. Li, S. M. Siniscalchi, I-F. Chen, J. Wu and C.-H. Lee, "Rapid Adaptation for Deep Neural Networks Through Multi-task Learning," INTERSPEECH 2015

(13) Z. Huang, J. Li, S. M. Siniscalchi, I-F. Chen, C. Weng and C.-H. Lee, "Feature Space Maximum A Posteriori Linear Regression for Adaptation of Deep Neural Networks," INTERSPEECH 2014

(14) Z. Huang, J. Li, C. Weng and C.-H. Lee, "Beyond Cross-entropy: Towards Better Frame-level Objective Functions for Deep Neural Network Training in Automatic Speech Recognition," INTERSPEECH 2014

(15) Y.-C. Cheng, V. Hautamki, Z. Huang, K. Li and C.-H. Lee, “An I-vector Based Descriptor for Alphabetical Gesture Recognition,” ICASSP 2014

(16) K. Li, Z. Huang, Y.-C. Cheng and C.-H. Lee, “A Maximal Figure-of-merit Learning Approach to Maximizing Mean Average Precision with Deep Neural Network Based Classifiers,” ICASSP 2014

(17) Z. Huang, C. Weng, K. Li, Y.-C. Cheng and C.-H. Lee, “Deep Learning Vector Quantization for Acoustic Information Retrieval,” ICASSP 2014

(18) Z. Huang, Y.-C. Cheng, K. Li, V. Hautamki and C.-H. Lee, “A blind segmentation approach to acoustic event detection based on i-vector,” INTERSPEECH 2014

(19) S. Oh, A. Perera, I. Kim, M. Pandey, K. Cannons, H. Hajimirsadeghi, A. Vahdat, G. Mori, B. Miller, S. McCloskey, Y.-C. Cheng, Z. Huang, C.-H. Lee, C. Xu, R. Kumar, W. Chen, J. Corso, L. Fei-Fei, D. Koller, V. Ramanathan, K. Tang, A. Joulin and A. Alahi, “TRECVID 2013 GENIE: Multimedia Event Detection and Recounting,” TREC Video Retrieval Evaluation Workshop 2013

(20) A. Perera, S. Oh, M. Pandey, T. Ma, A. Hoogs, A. Vahdat, K. Cannons, G. Mori, S.McCloskey, B. Miller, S. Venkatesh, P. Davalos, P. Das, C. Xu, J. Corso, R. Srihari,I. Kim, Y.-C. Cheng, Z. Huang, C.-H. Lee, K. Tang, L. Fei-Fei and D. Koller, “TRECVID 2012 GENIE: Multimedia Event Detection and Recounting,” TREC Video Retrieval Evaluation Workshop 2012